JAPAN PATENT OFFICE

This is to certify that the annexed is a true copy of the following application as filed with this office.

Date of Application: March 25, 2003
Application Number: JP 2003-082880
Applicant(s): BROTHER KOGYO KABUSHIKI KAISHA

[Title of Document]	Application for Patent
[Reference Number]	BRO2139
[Date of Filing]	March 25, 2003
[To]	Commissioner, Patent Office
[International Patent Code]	G06F 13/10
[Inventor]	
[Address]	c/o BROTHER KOGYO KABUSHIKI KAISHA No.15-1, Naeshiro-cho, Mizuho-ku, Nagoya-shi, Japan
[Name]	Tomohiro SUZUKI
[Address]	c/o BROTHER KOGYO KABUSHIKI KAISHA No.15-1, Naeshiro-cho, Mizuho-ku, Nagoya-shi, Japan
[Name]	Hiromi MORI
[Applicant]	
[Identification Number]	000005267
[Name]	BROTHER KOGYO KABUSHIKI KAISHA
[Agent]	
[Identification Number]	100082500
[Patent Attorney]	
[Name]	Tsutomu ADACHI
[Telephone Number]	052-231-7835
[Appointed Agent]	
[Identification Number]	100109195
[Patent Attorney]	
[Name]	Katsunori MUTO
[Designation of Charge]	
[Ledger No. for Prepayment]	007102

[Amount of Payment]	¥ 21,000	
[Lists of Document Attached]		
[Title of Document]	Specification	1
[Title of Document]	Drawings	1
[Title of Document]	Abstract	1
[Registration Number of General Power Attorney]	9006582	
[Registration Number of General Power of Attorney]	0018483	
[Necessity of Proofs]	Yes	

[Document Name] Specification

[Title of the Invention] Uninstall System, Uninstall Method, and Program

[Claims]

[Claim 1]

An uninstall system comprising:

an installing means for registering a function-specific device specification data, which is data to enable an operating system to specify a device as a device having a single function, in association with a device driver specification data, which is data for specifying a device driver as a program for controlling the device to execute the single function, in a device registration area managed by the operating system; and for enabling the operating system to find the function-specific device specification data recorded in the device registration area by an installation process for recording the function-specific device specification data in the device registration area in association with an input/output interface specification data, which is data for specifying an input/output interface used for performing communications to control the device, so as to execute the single function possessed by the device by controlling the device with the device driver specified by the device driver specification data associated with the function-specific device specification data via the

input/output interface specified by the input/output interface specification data associated with the function-specific device specification data; and

an uninstalling means for executing an uninstallation process to remove the input/output interface specification data associated with the function-specific device specification data from the device registration area and to remove the device driver specification data associated with the function-specific device specification data from the device registration area in order to prevent the operating system from controlling the device to execute the single function,

the uninstall system being characterized in further comprising:

an uninstall request receiving means for performing a process to receive an uninstall request from a user in a device unit, wherein:

the installing means performs a process as the installation process to store the function-specific device specification data or the device driver specification data in a storing means in association with device specification data, which is data indicating as a single device in the device unit; and

the uninstalling means performs an uninstallation process as the uninstallation process based on the function-

specific device driver specification data or the device driver specification data stored in the storing means in association with the device specification data when an uninstall request in a device unit is received from a user by the uninstall request receiving means.

[Claim 2]

The uninstall system according to claim 1, characterized in that:

the uninstallation process based on the function-specific device driver specification data or the device driver specification data is a process to remove entries of the input/output interface specification data and the device driver specification data stored in the device registration area in association with the function-specific device specification data stored in the storing means in association with the device specification data, or a process to remove entries of the input/output interface specification data and the device driver specification data stored in a device registration area managed by the operating system itself in association with the device driver specification data.

[Claim 3]

The uninstall system according to claim 1 or 2, characterized in further comprising:

an installation procedure storing means for storing

data as installation procedure including data indicating that the device driver must be installed for each function possessed by the device, wherein

the installing means determines a device driver that must be installed for enabling the operating system to control the device to execute the single function according to the installation procedure stored in the installation procedure storing means and installs the device driver.

[Claim 4]

The uninstall system according to any of claims 1 to 3, characterized in that:

the uninstall request receiving means receives an uninstall request from the user by prompting the user to select a device unit as the target of uninstallation; and

the uninstalling means performs the uninstallation process for the device unit selected by the user as the target of uninstallation.

[Claim 5]

The uninstall system according to claim 4, characterized in that:

the uninstalling means performs the uninstallation process for all of a plurality of device units when the user selects the plurality of device units as the target of uninstallation.

[Claim 6]

The uninstall system according to claim 4 or 5, characterized in that:

the installing means performs a process to store a device unit identification data, which is data enabling the user to identify the device unit, in the storing means in association with the device specification data; and

the uninstall request receiving means performs a process to display the device unit identification data stored in the storing means so as to prompt the user to select a device unit as the target of uninstallation based on the displayed device unit identification data.

[Claim 7]

The uninstall system according to claim 6, characterized in that:

the input/output interface specification data is used as the device unit specification data.

[Claim 8]

The uninstall system according to claim 7, characterized in that:

the uninstall request receiving means reads the device driver specification data corresponding to the selected device unit from the storing means and displays the same.

[Claim 9]

The uninstall system according to any of claims 4 to 8, characterized in that:

the installing means further performs a process to store the device unit identification data and a model specification data, which is data specifying the model of the device, in the storing means in association with each other; and

the uninstall request receiving means performs a process to display the model specification data stored in the storing means to prompt the user to select a model specification data as the target of uninstallation, and sets each device unit identified by all device unit identification data stored in the storing means in association with the model specification data selected by the user as the target of uninstallation as a device unit as the target of uninstallation.

[Claim 10]

The uninstall system according to claim 9, characterized in that:

the uninstall request receiving means displays in association with the device unit identification data corresponding to the model specification data.

[Claim 11]

The uninstall system according to any of claims 1 to 10, characterized in that:

the functions of the single device are at least one of a printer function, an image reading function, and a fax

communication function.

[Claim 12]

The uninstall system according to any of claims 1 to 10, characterized in that:

the single device is a multifunction device.

[Claim 13]

An uninstall method that replaces each means in the uninstall system according to any of claims 1 to 14, except the storing means, with step and directs a computer to execute each step.

[Claim 14]

An uninstall method comprising:

an installing step for registering a function-specific device specification data, which is data to enable an operating system to specify a device as a device having a single function, in association with a device driver specification data, which is data for specifying a device driver as a program for controlling the device to execute the single function, in a device registration area managed by the operating system; and for enabling the operating system to find the function-specific device specification data recorded in the device registration area by recording the function-specific device specification data in the device registration area in association with an input/output interface specification data, which is data for specifying

an input/output interface used for performing communications to control the device, so as to execute the single function possessed by the device by controlling the device with the device driver specified by the device driver specification data associated with the function-specific device specification data via the input/output interface specified by the input/output interface specification data associated with the function-specific device specification data; and

an uninstalling step for executing an uninstallation process to remove the input/output interface specification data associated with the function-specific device specification data from the device registration area and to remove the device driver specification data associated with the function-specific device specification data from the device registration area in order to prevent the operating system from controlling the device to execute the single function,

the uninstall method being characterized in further comprising:

an uninstall request receiving step for performing a process to receive an uninstall request from a user in a device unit, wherein:

the installing step performs a process as the installation process to store the function-specific device specification data or the device driver specification data

in a storing means in association with device specification data, which is data indicating as a single device in the device unit; and

the uninstalling step performs uninstallation process as the uninstallation process based on the function-specific device driver specification data or the device driver specification data stored in the storing means in association with the device specification data when an uninstall request in a device unit is received from a user by the uninstall request receiving step.

[Claim 15]

A program for directing a computer to execute a process as the uninstall system of any of claims 1 to 12.

[Detailed Description of the Invention]

[0001]

[Technical Field of the Invention]

The present invention relates to an uninstall system, an uninstall method, and a program for uninstalling device drivers, such as a scanner driver or a printer driver, and particularly to an uninstall system, an uninstall method, and a program capable of uninstalling a plurality of device drivers at the same time.

[0002]

[Prior Art]

Conventional install systems well known in the art

install application software program on an operating system. This type of install system performs a process to copy each component of application software program having one or more components (programs and data) to an appropriate location and to set data required for executing the application software program such that the application software program is executable under the control of the operating system.

[0003]

Also, an install system known in the art copies device drivers needed to use a device under control of an operating system to a suitable location under the administration of the operating system, and records data required for operating the device drivers in the operating system.

[0004]

However, when using a multifunction device provided with a printer function, a scanner function, a fax function, and the like (in this type of multifunction device, a copy function can be implemented by combining the scanner function and the printer function) from a personal computer, it is necessary to install each of a printer driver, a scanner driver, a fax driver, and an application program relating to usage of the multifunction device on the operating system in the personal computer (see Patent document 1, for example).

[0005]

Also, when installing a plurality of software programs to use a single device in this manner, time and effort is required to install each software program individually.

Therefore, an install system well known in the art was developed to automatically install all software programs required for using the functions of the multifunction device by combining the plurality of software programs in a single package and storing this package on a storage medium, such as a CD-ROM. During installation, each software program is extracted individually from the package and installed in sequence. Another install system well known in the art prompts the user to select and input settings for device drivers and applications during the installation process and performs settings based on the selections inputted by the user.

[0006]

For example, When installing device drivers (Scanner Driver A and Printer Driver A) using the install system provided in an operating system commonly used on personal computers, a first multifunction device (Device 1) having the model name "Model A" and provided with a printer function and a scanner function is connected to a default port of the personal computer, Port A, and the install system enters settings data in registries, which are areas in the operating system for storing settings data, as shown

in Fig. 20(a).

[0007]

That is, as shown in Fig. 20(a), the install system creates a key 111 of Device 1 for a system registry (scanner) 110. The install system sets "Scanner Driver A" as a driver specification data 111a, "Scanner Interface A" as an input/output interface specification data, and "Scanner A-1" as a scanner name 111c and stores this data under the key 111. The input/output specification data for the scanner can conceivably be the name of the STI driver, which is an interface driver, and the IP address of the scanner with which the STI driver should communicate. Then, it is determined whether or not the Scanner Driver A is already stored in a scanner driver storage area of the operating system. If not stored, then the Scanner Driver A stored on the CD-ROM or the like is copied to the scanner driver storage area of the operating system, and a request is issued to the operating system to set the Scanner Driver A to Port A.

[0008]

Similarly, as shown in Fig. 20(a), the install system creates a key 121 of the Device 1 for the system registry (printer) 120. The install system sets "Printer Driver A" as a driver specification data 121a, "Printer Interface A" as an input/output interface specification data, and

"Printer A-1" as a printer name 121c and stores this data under the key 121. The input/output interface specification data for the printer can conceivably be a port name or the like specifying a port, which is a logical interface recorded in the operating system by a port driver, which is the interface driver. It is then determined whether or not Printer Driver A is already stored in the printer driver storage area of the operating system. If not stored, then the install system copies the Printer Driver A stored in the CD-ROM or the like to the printer driver storage area of the operating system and issues a request to the operating system to set Printer Driver A to Port A.

[0009]

Further, as shown in Fig. 20(a), the install system sets "Model A" as a model name 131 and stores this data under a vendor registry 130.

As a result, if the user clicks on a folder (scanner) managed by the operating system, the operating system displays a folder (scanner) window 150 shown in Fig. 20(b) on a display device of the personal computer based on the settings data. Displayed within the folder (scanner) window 150 are a scanner name "Scanner A-1" and its icon 151a. When the user clicks on this icon, the operating system executes a process to acquire the status of the first multifunction device connected to Port A that corresponds to

Scanner A-1 from Port A, using Scanner Driver A and Scanner Interface A and to display the status. The operating system also displays a dialog box enabling to view and modify settings for Scanner A-1, to store settings modified by the user in the displayed dialog box, and to output to the first multifunction device via Port A using Device Driver A. Further, if the operating system receives a scan request from an application program specifying Scanner A-1, the operating system can control the first multifunction device via Port A and according to Scanner Driver A and Scanner Interface A to execute a scan operation.

[0010]

Similarly, when the user clicks on the folder (printer) managed by the operating system, the operating system displays a folder (printer) window 160 shown in Fig. 20(b) on the display device of the personal computer based on the settings data. Displayed within the folder (printer) window 160 are a printer name "Printer A-1" and its icon 161a. When the user clicks on this icon, a process is executed to acquire the status of the first multifunction device connected to Port A that corresponds to Printer A-1 from Port A, using Printer Driver A and Printer Interface A, and to display the status. The operating system also displays a dialog box enabling to view and modify settings for Printer A-1, to store settings modified by the user in

the displayed dialog box, and to output to the first multifunction device via Port A using Printer Driver A and Printer Interface A. Further, if the operating system receives a print request from an application program specifying Printer A-1, the operating system can control the first multifunction device via Port A using Printer Driver A and Printer Interface A to execute a print operation.

[0011]

Further, when a second multifunction device (Device 2) of the same model as the first multifunction device (model name is "Model A") is connected to Port B of the personal computer to which the first multifunction device is also connected, the user inputs data to the install system indicating connection to Port B. Since Scanner Driver A and Printer Driver A are already installed, the install system skips the process to copy these device drivers and executes a process to set registries and to issue a request to the operating system to set each device driver to Port B. As a result, as shown in Fig. 21(a), a key 112 for Device 2 is added to the system registry (scanner) 110. The install system also sets "Scanner Driver A" as a driver specification data 112a, "Scanner Interface B" as an input/output interface specification data, and "Scanner A-2" as a scanner name 112c under the key 112. Similarly, a key 122 for Device 2 is added to the system registry (printer)

120, and the install system sets "Printer Driver A" as a driver specification data 122a, "Printer Interface B" as an input/output interface specification data, and "Printer A-2" as a printer name 122c under the key 122.

[0012]

In this way, a separate connection point (port in this example) can be added and set in addition to the connection point set when the device driver was first installed. The settings data for the device driver is managed for each connection point. In the above-described example, as shown in Fig. 21(b), icons 151a and 151b are created for each connection point, enabling the user to select a connection point corresponding to one of the icons from the application program or the operating system in order to select the device to be used.

[0013]

With this install system, software programs required for using a device can be easily installed, even when a plurality of software programs must be installed to use a single device, such as a multifunction device. Accordingly, it is possible to quickly use the device from a personal computer.

[0014]

Further, an uninstall system for uninstalling a plurality of specified ones of a plurality of installed

application programs at once is known in the art. For example, when a plurality of application programs have been installed, a process is performed to delete all individual components of the application programs, and to delete all settings data required for operating the application programs.

[0015]

[Patent Document 1]

Japanese unexamined patent application publication No. HEI-10-97485

[0016]

[Problems to Be Solved By the Invention]

However, regarding device drivers, since there has not been a technique to specify the association of device drivers after installation, there has been a problem that a plurality of device drivers cannot be uninstalled all at once.

[0017]

For example, when a printer driver, a scanner driver, and a fax driver are installed on an operating system, data required to operate each device driver is only set in the operating system during installation of each device driver, and it is not possible to specify the association of these device drivers. Thus, the device drivers cannot be uninstalled all at once.

[0018]

In view of the foregoing, it is an object of the present invention to provide an uninstall system enabling a user to easily uninstall a plurality of device drivers for a device all at once when the plurality of device drivers is necessary to operate the single device.

[0019]

[Means to Solve the Problems and Effects of the Invention]

The uninstall system according to claim 1, which is provided for solve the above problems, includes: an installing means for registering a function-specific device specification data, which is data to enable an operating system to specify a device as a device having a single function, in association with a device driver specification data, which is data for specifying a device driver as a program for controlling the device to execute the single function, in a device registration area managed by the operating system, and for enabling the operating system to find the function-specific device specification data recorded in the device registration area by an installation process for recording the function-specific device specification data in the device registration area in association with an input/output interface specification data, which is data for specifying an input/output interface used for performing communications to control the device, so

as to execute the single function possessed by the device by controlling the device with the device driver specified by the device driver specification data associated with the function-specific device specification data via the input/output interface specified by the input/output interface specification data associated with the function-specific device specification data; and an uninstalling means for executing an uninstallation process to remove the input/output interface specification data associated with the function-specific device specification data from the device registration area and to remove the device driver specification data associated with the function-specific device specification data from the device registration area in order to prevent the operating system from controlling the device to execute the single function, the uninstall system further includes an uninstall request receiving means for performing a process to receive an uninstall request from a user in a device unit, wherein: the installing means performs a process as the installation process to store the function-specific device specification data or the device driver specification data in a storing means in association with device specification data, which is data indicating as a single device in the device unit; and the uninstalling means performs uninstallation process as the uninstallation process based on the function-specific device driver

specification data or the device driver specification data stored in the storing means in association with the device specification data when an uninstall request in a device unit is received from a user by the uninstall request receiving means.

[0020]

Hence, even when a plurality of device drivers is required for using a single device, a user can uninstall the plurality of device drivers for the device all at once simply by indicating a device unit. Accordingly, it is possible to provide a convenient uninstall system that does not inconvenience the user.

[0021]

Note that the uninstallation process based on the function-specific device driver specification data or the device driver specification data may be a process to remove entries of the input/output interface specification data and the device driver specification data stored in the device registration area in association with the function-specific device specification data stored in the storing means in association with the device specification data, or a process to remove entries of the input/output interface specification data and the device driver specification data stored in a device registration area managed by the operating system itself in association with the device

driver specification data, as described in claim 2.

[0022]

In this way, the input/output interface specification data and device driver specification data recorded in the operating system during installation can be removed by specifying the device specification data for a device to be uninstalled.

[0023]

When installing a device driver, it is possible to determine whether or not it is necessary to install the plurality of device drivers for controlling a single device, for example, by receiving instructions inputted by the user. However, as described in claim 3, it is preferable that an installation procedure storing means for storing data as an installation procedure including data indicating that the device driver must be installed for each function possessed by the device be provided and that the installing means determine a device driver that must be installed for enabling the operating system to control the device to execute the single function according to the installation procedure stored in the installation procedure storing means and install the device driver.

[0024]

Further, the device unit that is the target of uninstallation may be automatically detected by detecting

the connection status of the device, for example. However, as described in claim 4, for example, the uninstall request receiving means may receive an uninstall request from the user by prompting the user to select a device unit as the target of uninstallation, and the uninstalling means may perform the uninstallation process for the device unit selected by the user as the target of uninstallation. With this construction, the user can specify a device to be the target of uninstallation.

[0025]

As described in claim 5, the uninstallation process may be performed for all of a plurality of device units when the user selects the plurality of device units as the target of uninstallation. With this construction, the user can have a plurality of selected devices uninstalled altogether.

[0026]

Also, as described in claim 6, the installing means may perform a process to store a device unit identification data, which is data enabling the user to identify the device unit, in the storing means in association with the device specification data, and the uninstall request receiving means may perform a process to display the device unit identification data stored in the storing means so as to prompt the user to select a device unit as the target of uninstallation based on the displayed device unit

identification data. With this configuration, since device unit identification data is displayed when selecting a device unit that is the target of uninstallation, the user can easily select a device unit as the target of uninstallation. The input/output interface specification data may be used as the device unit identification data as described in claim 7, for example.

[0027]

As described in claim 8, the uninstall request receiving means may read the device driver specification data corresponding to the selected device unit from the storing means and display the same. In this way, the user can easily learn which device driver is the target of uninstallation.

[0028]

As described in claim 9, the installing means may further perform a process to store the device unit identification data and a model specification data, which is data specifying the model of the device, in the storing means in association with each other, and the uninstall request receiving means may perform a process to display the model specification data stored in the storing means to prompt the user to select a model specification data as the target of uninstallation, and set each device unit identified by all device unit identification data stored in

the storing means in association with the model specification data selected by the user as the target of uninstallation as a device unit as the target of uninstallation. In this way, uninstallation can be performed for each model. Further, if the user wishes to uninstall all device drivers for a specific model, for example, the user can uninstall all device drivers for a specific model simply by specifying the model.

[0029]

Also, when receiving the uninstall request, it is preferable to display in association with the device unit identification data corresponding to the model specification data, as described in claim 10. With this construction, since the user can learn which model the device unit identification data is for, the user can easily select a device unit or model specification data as target for uninstallation.

[0030]

Note that the functions of the single device may be at least one of a printer function, an image reading function, and a fax communication function, as described in claim 11. With this configuration, it is possible to apply the present invention to a system in which the operating system directs a device via an input/output interface to process image data. Also, the device may be any device that requires

installation of a plurality of device drivers, for example. For example, the device may be a multifunction device, as described in claim 12. With this configuration, the present invention can be applied to a system in which the operating system controls a device by controlling a device with a device driver for each function in order to implement a single function possessed by the device. Further, MS Windows (registered trademark) for example may be used as the operating system; the scanner registry or printer registry for example may be used as the device registration area; the scanner registry key or printer registry key for example may be used as the function-specific device specification data; the port name and STI driver name + address for example may be used as the input/output interface specification data; the network port + address or the USB port for example may be used as the input/output interface; the scanner driver name or printer driver name for example may be used as the device driver specification data; and the installer registry for example may be used as the storing means. Further, install may be a process for entering in the registry for example; a key name for example may be the device specification data; uninstall may be a process for removing entries from the registry for example; the installation procedure storing means may be an inf file for example; the device unit identification data may be a

port name for example; and the model specification data may be a model name for example.

[0031]

Based on the invention disclosed in each claim described above, an uninstall system may be configured as follows, for example.

While the uninstall system according to claims 1 to 12 have been described in the apparatus claim format, the same effects can be achieved when the invention is described in a method format as the uninstall method as described in claim 13. For example, the uninstall system according to claim 14 is an example of the invention described as the method invention corresponding to claim 1. Claims 1 to 12 can similarly be converted to an invention of the method category.

[0032]

Further, as described in claim 16, it is possible to configure as a program for directing a computer to execute process of the uninstall system according to any of claims 1 to 12. Such a program may be stored on a storage medium that can be read by a computer, such as a flexible disk, an optical disc (CD-ROM or DVD-ROM, for example), a hard disk, a ROM, a RAM, or the like and loaded and executed on the computer when required, or may be loaded via a network and executed.

[0033]

[Embodiment of the Invention]

Next, an embodiment of the present invention will be described while referring to the accompanying drawings. Note that the embodiment of the present invention is not limited to the following embodiment, and it would be apparent that many modifications and variations are possible within the technical field of the present invention.

[0034]

Fig. 1 is a diagram showing the construction of a multifunction-device employing system 100 including a personal computer 1 functioning as an uninstall system of the present embodiment, a CD-ROM 10 storing various programs including an installer, an uninstaller, a device driver, and the like for controlling the computer to function as an uninstall system, and multifunction devices 2 (2a to 2c).

[0035]

The personal computer 1 is a common personal computer including a CPU 11, a RAM 12, a CD-ROM drive 13, a hard disk drive 14, a parallel port 15, a network interface 16, and the like. The personal computer 1 can run an operating system stored on the hard disk drive 14 and display various data on a display device not shown in the drawings. Data can be selected and input through a mouse and a keyboard not shown in the drawings.

[0036]

The CPU 11 can read programs and data into the RAM 12 from the CD-ROM 10 inserted in the CD-ROM drive 13 and execute processes using the operating system stored on the hard disk drive 14.

A printer cable 4 is connected to the parallel port 15 of the personal computer 1, and the other end of the printer cable 4 is connected to the multifunction device 2a, enabling to communicate electrically with the personal computer 1. The network interface 16 of the personal computer 1 is connected to a network 3. The multifunction device 2b and the multifunction device 2c are connected to the network 3.

[0037]

Upon receiving signals from the CD-ROM drive 13 indicating that the CD-ROM 10 has been inserted, the CPU 11 controls the CD-ROM drive 13 to read and execute the install program stored on the CD-ROM 10.

The installation process implemented on the personal computer 1 when the CPU 11 executes the install program will be described with reference to Figs. 2 through 10.

[0038]

Fig. 2 is a flowchart showing steps in the installation process. Fig. 3 is a flowchart showing steps in a registry creation process in S100 of Fig. 2. Fig. 4 is

a flowchart showing steps in a local registry creation process in S400 of Fig. 2. Note that the registry is a settings storing file that is managed by the operating system and is stored on the hard disk drive 14.

[0039]

The installation process in Fig. 2 begins by performing the registry creation process in S100. This process is a process shown in Fig. 3.

First, a variable RegNum is created to indicate the registry number (S110) and is initialized such that RegNum=1 (S120). Then, the model name data is acquired from an ini file stored on the CD-ROM 10 and a variable szModel is set to this data (S130). All registries existing under "Registry 1" + szModel are listed, where "Registry 1" is the base path in the registry incorporated in the installer. An example of the registry is shown in Fig. 5. For example, "¥HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥" is added as "Registry 1", and "BrMF1" is added as szModel, and then a list of registries under "¥HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥BrMF1" is acquired and set as RegList (S140). Then, the first data entry in RegList is acquired as acquired registry data (S150). For example, if the registry is that shown in Fig. 5, "1" under "¥HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥BrMF1" is acquired as the acquired registry data. If an installation

has never been performed in relation to this driver, for example, then a key of "¥HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥BrMF1" itself does not exist in the registry, and thus it is not possible to acquire the first data entry in RegList.

[0040]

In S160, it is determined whether or not the first data entry in RegList was acquired in S150. If acquired (S160: YES), then the process advances to S170. However, if not acquired (S160: NO), then the process advances to S220.

[0041]

In S170, RegNum is set to the acquired registry data. For example, in the case of Fig. 5 described above, RegNum =1.

Then, it is attempted to acquire the next data entry in RegList (S180), and it is determined whether the data was acquired (S190). If acquired (S190: YES), then the process advances to S200. If not acquired (S190: NO), the process advances to S220.

[0042]

In S200, it is determined whether or not the acquired registry data acquired in S180 is larger than RegNum. If so (S200: YES), then the process advances to S210, where RegNum is set to the value of the acquired registry data, and subsequently the process returns to S180. However, if not

(S200: NO), then the process directly returns to S180. In other words, the greatest value among the acquired registry data is searched for. In the example of Fig. 5 described above, the largest value is "3".

[0043]

In S220, it is determined whether a key for RegNum exists in the registry. If exists (S220: YES), then in S230 RegNum is incremented by "1". However, if does not exist (S220: NO), then the process advances to S240. As a result, in the example of Fig. 5 described above, RegNum is set to 3+1=4.

[0044]

In S240, a data storage registry is set to "Registry1¥szModel¥RegNum." As a result, in the example of Fig. 5, "¥HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥BrMF1¥4" is set as the data storage registry.

Subsequently, the process advances to S310 in Fig. 2.

[0045]

In S310, it is determined whether the device being installed is a local connection or a network connection based on input by a user on a selection dialog box. If it is a local connection (S310: Local), then the process advances to S400, and the process shown in Fig. 4 is executed. If it is a network connection (S310: Network), then the process advances to S320.

[0046]

If a local registry already exists, then the process of Fig. 4 (process in S400) is a process for resetting the registry as the data storage registry. In the multifunction-device employing system according to the present embodiment, this process ensures that only one multifunction device can be connected to the personal computer 1 as a local connection.

[0047]

In S410 of Fig. 4, a subkey located directly under Registry 1 is acquired, and RegListLocal1 is set to the subkey. Then, it is attempted to acquire the first data entry under RegListLocal1 as the acquired registry data (S420), and it is determined whether or not a data entry was acquired (S430). If acquired (S430: YES), then the process advances to S440. If not acquired (S430: NO), then the process advances to S320 in Fig. 2.

[0048]

In S440, RegSub1 is set to the acquired registry data.

In S450, a list of registries under Registry1¥RegSub1 is acquired, and RegListLocal2 is set to this list.

In S460, it is attempted to acquire the first data entry in RegListLocal2, and it is determined whether an entry was acquired (S470). If acquired (S470: YES), then it is set as the acquired registry data of

RegSub2=RegListLocal2 (S485), and the process advances to S490. However, if not acquired (S470: NO), then the process advances to S480.

[0049]

In S480, it is attempted to acquire the next data entry in RegListLocal1, and the process returns to S430.

In S490, the value of a value name (PortName) under RegSub2 is acquired, and szPort is set to this value. Then, it is determined whether or not szPort is "Local" (S500). If szPort="Local" (S500: YES), then the process advances to S520. However, if szPort ≠ "Local" (S500: NO), then the process advances to S510. In S510, it is attempted to acquire the next data entry in RegListLocal2, and the process returns to S470.

[0050]

In S520, the data storage registry is set to Registry1¥RegSub1¥RegSub2, and the process advances to S330 in Fig. 2. In this way, if a "Local" registry already exists, then the data storage registry is changed to that registry. However, if a "Local" registry does not exist (S430: NO), then the data storage registry remains set to the registry created in S100.

[0051]

In S320, an IP address is set. That is, data for devices connected to the network interface 16 via the

network 3 is acquired, and the IP addresses and model names from among this data are displayed in a dialog box, such as that shown in Fig. 6, so as to prompt the user to make a selection. Then, an IP address selected by the user is set as an IP address to be installed.

[0052]

In S330, the driver information file (inf file) shown in Fig. 7 is transferred to the operating system, and the driver files (printer driver, scanner driver, and fax driver) are copied to their respective driver storage areas for the operating system on the hard disk drive 14, and a request to the operating system to register these drivers is issued. In other words, the operating system is directed to register settings data in the system registry for each driver (store settings data to the keys in the system registry). Fig. 8 shows an example of settings data for the scanner driver registered in the system registry. Directly after the registration, the settings data that the operating system has registered to the registry key is read from the registry key, and the data is written to the registry in the following step (S340). One method for identifying the registry key directly after the operating system has stored the settings data is to request to store a prescribed value in the registry key in which the settings data is stored when requesting the operating system to register the drivers

and to detect the registry key that stores the prescribed value when reading the settings data. For example, the scanner driver registry key name (TwainClassData) ("0047" in the drawing) returned by the operating system shown in Fig. 8 is stored in the data storage registry. In addition, an uninstaller program and icon data are copied from the CD-ROM 10 onto the hard disk drive 14.

[0053]

In S340, data required for uninstallation is recorded in the data storage registry. The required data includes a PC-FAX name (FaxName), a port name (PortName), a printer name (PrinterNameHBP), the scanner driver registry key name (TwainClassData) mentioned earlier, a TWAIN folder name (TwainDirName), and a TWAIN DS name (TwainDSName). Fig. 5 shows an example of data recorded in the registry of a local port, while Fig. 9 shows an example of data recorded in the registry of a network port. Then, in S350, the port name or "USB," "LPT," and the like are recorded as comments in a printer registration key of the system registry (printer, fax). In this way, installation is performed such that the IP address selected in S320 is included in the port name (PortName).

[0054]

When the installation process is completed, the registry, files (printer driver, scanner driver, and fax

driver), the uninstaller, and the icon data are stored on the hard disk drive 14 as shown in Fig. 10.

In this state, the operating system can control a device via the device driver according to commands from an application program or the like. Accordingly, the user can use the multifunction device 2 from the personal computer 1 by controlling the multifunction device 2 via a port according to the scanner driver and the scanner interface or the printer driver and the printer interface, as described in the section on the prior art (see Figs. 20 and 21).

[0055]

Note that the steps of the installation process described above may also be incorporated in an install program (corresponding to the install procedure storing means). For example, the above installation process may be performed using an installation procedure file (corresponding to the install procedure storing means) describing the procedure for the installation process described above and an install program for performing a process according to the description in the installation procedure file.

[0056]

If the user no longer needs to use the multifunction device 2, the uninstaller stored on the hard disk drive 14 is executed.

An uninstallation process implemented on the personal computer when the CPU 11 executes the uninstaller will be described with reference to Figs. 11 through 14.

[0057]

Figs. 11 to 14 are flowcharts showing steps in the uninstallation process. Fig. 14(a) is a flowchart showing steps in an uninstall list 1 creation process in S600 of Fig. 11. Fig. 14(b) shows examples of "Registry 1," "Registry 2," and "Registry 3" in the process of Fig. 14(a).

[0058]

In S600 of Fig. 11, an uninstall list 1 creation process is executed. This process will be described in detail with reference to Fig. 14.

In S610 of Fig. 14, Registry 1, that is a list of registries under "¥HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥" shown in Fig. 9 is acquired and set as RegList1. Then, it is attempted to acquire the first character array data in RegList1 (S620), and it is determined whether or not data was acquired (S630). If acquired (S630: YES), then the process advances to S640. However, if not acquired (S630: NO), then the process advances to S810 in Fig. 11.

[0059]

In S640, Registry 2 is set to Registry 1 + SubKey1. In S650, a list of registries under Registry 2 is acquired and set as RegList2. In S660, it is attempted to acquire

the top subkey in RegList2, and SubKey2 is set to the acquired character array data. In S670, it is determined whether data was acquired in S660. If acquired (S670: YES), then the process advances to S690. However, if not acquired (S670: NO), then the process advances to S680.

[0060]

In S680, it is attempted to acquire the next character array data in RegList1, and SubKey1 is set to the acquired character array data. Then, the process returns to S630.

In S690, Registry 3 is set to Registry 2 + SubKey2. In S700, Registry 3 is added to the uninstall list 1. In S710, it is attempted to acquire the next character array data in RegList2, and SubKey2 is set to the acquired character array data. Then, the process returns to S670.

[0061]

Through the uninstall list 1 creation process, the uninstall list 1 shown in Fig. 15(a) is obtained from the registry shown in Fig. 9, for example.

Then, in S810 of Fig. 11, an uninstall type selection dialog box shown in Fig. 16 is displayed. The uninstall type selection dialog box enables to select the connection point of the device (multifunction device 2) that the user wishes to uninstall. For example, when uninstalling the multifunction device 2a connected to the parallel port 15, the user selects "Local Interface." When uninstalling the

multifunction device 2b or the multifunction device 2c on the network 3 connected to the network interface 16, the user selects "Network Interface." To uninstall all of these devices, the user selects "All." Then, the user clicks on the Next button. The selected one is set as the uninstall type.

[0062]

In S820, it is determined whether the Next button has been selected. If the Next button has been selected, the process advances to S830.

In S830, the selected uninstall type is stored in the RAM 12. In S840, it is attempted to acquire the first data entry in the uninstall list 1, and it is determined whether or not the data entry was acquired (S850). If acquired (S850: YES), then the process advances to S860. However, if not acquired (S850: NO), then the process advances to S920 of Fig. 12.

[0063]

In S860, the port name is acquired from the SubKey2. In S870, the uninstall type is determined based on the uninstall type stored in the RAM 13 in S830. If the uninstall type is "Local," then the process advances to S890. If the type is "Network," then the process advances to S880. If the type is "All," then the process advances to S900.

[0064]

In S880, it is determined whether the port name acquired in S860 is not "Local." If the port name is not "Local" (S880: YES), then the process advances to S900, but if "Local" (S890: NO), the process advances to S895.

In S890, it is determined whether the port name acquired in S860 is "Local." If "Local" (S890: YES), then the process advances to S900, but if not "Local" (S890: NO), then the process advances to S895.

[0065]

In S895, it is attempted to acquire the next data entry in the uninstall list 1, and the process returns to S850.

In S900, the data is registered in an uninstall list 2. In S910, after part of the SubKey2 is changed to the port name, the uninstall list 2 is registered in an uninstall list 3 and the list dialog box. Then, the process advances to S895.

[0066]

As a result of this process, only registry keys of the uninstall type selected in the dialog box of Fig. 15 are stored in the uninstall list 2. In third uninstall list 3, the number part of the SubKey2 in the registry key of the uninstall list 2 is replaced with the port name. In the case of the registry shown in Fig. 9, for example, values in Fig. 15(a) described above are stored in the uninstall list

1. If the user selects "Local" as the uninstall type and if the only entry in the uninstall list 1 having "Local" as the port name is the entry having "2" as Subkey 2 of "¥HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1," then uninstall list 2 and uninstall list 3 are created as shown in Fig. 15(b).

[0067]

Further, when All is selected as the uninstall type, for example, uninstall list 2 and uninstall list 3 are created as shown in Fig. 15(c). The elements in each list are configured to be referenced by index values.

[0068]

In S920 of Fig. 12, a list dialog box is displayed as shown in Fig. 17. Note that the list dialog box of Fig. 17 shows an example for when "All" was selected in the dialog box of Fig. 16. When "Local Interface" or "Network Interface" is selected, only the corresponding devices are displayed. In this list dialog box, each node of the tree portion under BrMFL in the registry tree of Fig. 9 is displayed along with a checkbox. When a node corresponding to a leaf of the tree is clicked, the checkbox is toggled from a current state. When a parent node of the tree is clicked, the current state of the parent node is toggled, and states of all checkboxes of the tree under the parent node are also toggled. Checked nodes are treated as being

selected.

[0069]

Of the registry values (content in the Data column in Fig. 9) corresponding to the node selected in the tree (the node toggled in Fig. 17), values for PortName, PrinterNameHBP, TwainDSName, and FaxName are sequentially displayed in the Description area. Accordingly, the user can easily specify the target of uninstallation.

[0070]

In S930, a Description display process is executed. This process is shown in detail in Fig. 18. In S932, it is determined whether or not an element in the list dialog box of Fig. 17 has been clicked. If clicked (S932: YES), then the process advances to S933. If not clicked (S932: NO), then the process advances to S936.

[0071]

In S933, the index value of the element is acquired. In S934, Description data corresponding to the index value is acquired. In S935, the Description display of the list dialog box is switched to the Description data acquired in S934.

[0072]

Then, in S936, it is determined whether or not the Next button in the list dialog box has been selected. If selected (S936: YES), then the process is ended, and the

process advances to S940 in Fig. 12. However, if the Next button has not been selected (S936: NO), then the process returns to S932.

[0073]

Then, in S940 of Fig. 12, the existence of selections in the list dialog box is confirmed. In S950, items not selected in the list dialog box are deleted from the uninstall list 2. Then, in S955, it is attempted to acquire the first data entry in the uninstall list 2. In S960, it is determined whether a data entry was acquired. If acquired (S960: YES), then the process advances to S970. If not acquired (S960: NO), then the process advances to S1210 of Fig. 13.

[0074]

In S970, TWAIN DS Name is acquired from the registry indicated by the acquired registry key name. Then, the scanner driver is deleted using API of the operating system using this TWAIN DS name. Further, in S990, TWAIN directory name is acquired from the registry indicated by the acquired registry key name, and in S1000, the TWAIN directory is deleted. Further, in S1110, the scanner driver registry key is acquired from the registry indicated by the acquired registry key name, and the scanner driver registry is deleted. For example, the tree below "0047" in Fig. 8 is deleted. Then, in S1130, the port name is acquired from the

registry indicated by the acquired registry key name. In S1150, it is determined whether or not the port name is "Local." If the port name is "Local" (S1150: YES), then the process advances to S1160 to delete one with the printer comment "USB" recorded in S350 from the printer folder. In S1170, one with the printer comment "LPT" is deleted from the printer folder.

[0075]

However, in S1180, one with the printer comment being the port name is deleted from the printer folder.

In S1190, the deleted printer driver name is described in the file Brlst.ini, as shown in Fig. 19, for example.

[0076]

In S1200, the registries of uninstall list 2 are deleted. Then, the process returns to S960.

However, in S1210, the printer described in the Brlst.ini file is deleted.

According to the uninstall system of the present embodiment, a vendor registry for the multifunction devices 2 (shown in Figs. 5 and 9) is provided separately from the conventional system registry (shown in Figs. 8, 21, and 22). By storing, in the vendor registry, data identifying the plurality of device drivers, including the scanner driver, the printer driver, the fax driver, and the like, in association with one another, it is possible to easily

uninstall the plurality of device drivers at one time.

[0077]

Further, it is possible to uninstall a plurality of device drivers at one time based on specific unit indicated by the user. Accordingly, it is possible to provide a convenient uninstall system that is not troublesome for the user to operate.

[Brief Description of the Drawings]

[Fig. 1] An explanatory diagram showing the structure of an uninstall system and the like according to an embodiment

[Fig. 2] A flowchart showing steps in an installation process

[Fig. 3] A flowchart showing steps in a registry creation process in S100 of Fig. 2

[Fig. 4] A flowchart showing steps in a local registry creation process in S400 of Fig. 2

[Fig. 5] An explanatory diagram showing an example registry (value is for when port is local)

[Fig. 6] An explanatory diagram showing an example dialog box for selecting an IP address to be installed

[Fig. 7] An explanatory diagram showing an example of driver information file (inf file)

[Fig. 8] An explanatory diagram showing an example of settings data for the scanner driver registered in the

system registry

[Fig. 9] An explanatory diagram showing an example of registry (value is for when port is network)

[Fig. 10] An explanatory diagram showing data stored on the hard disk drive when the installation process has completed

[Fig. 11] A flowchart showing steps in an uninstallation process

[Fig. 12] A flowchart showing steps in a continuation of the uninstallation process of Fig. 11

[Fig. 13] A flowchart showing steps in a continuation of the uninstallation process of Fig. 12

[Fig. 14] A flowchart showing steps in an uninstall list 1 creation process in S600 of Fig. 11

[Fig. 15] An explanatory diagram showing examples of uninstall lists 1-3

[Fig. 16] An explanatory diagram showing an example of an uninstall type selection dialog box

[Fig. 17] An explanatory diagram showing an example of a list dialog box

[Fig. 18] An explanation diagram showing steps in a Description display process in S930 of Fig. 12

[Fig. 19] An explanatory diagram showing an example of Brlst.ini file

[Fig. 20] A first diagram for explaining conventional

installation and uninstallation

[Fig. 21] A second diagram for explaining conventional installation and uninstallation

[Explanation of the Numberings]

1	personal computer
2a-c	multifunction device
3	network
11	CPU
12	RAM
13	CD-ROM drive
14	hard disk drive
15	parallel port
16	network interface

[Document Name] Abstract

[Abstract]

[Object] To provide an uninstall system enabling a user to easily uninstall a plurality of device drivers for a device all at once when the plurality of device drivers is necessary to operate the single device

[Configuration] A vendor registry for a multifunction device that is different from a conventional system registry is created during installation. Data specifying a plurality of related device drivers, such as a scanner driver, a printer driver, and a fax driver, is stored in association with each other in the vendor registry. Accordingly, it is possible to uninstall the plurality of device drivers at once. Further, it is possible to uninstall a plurality of device drivers in a unit that a user has specified on a list dialog box shown in a drawing. Thus, it is possible to provide a convenient uninstall system that is not troublesome for the user to operate.

[Selected Drawing] Fig. 17

Document Name Drawings

Fig. 1

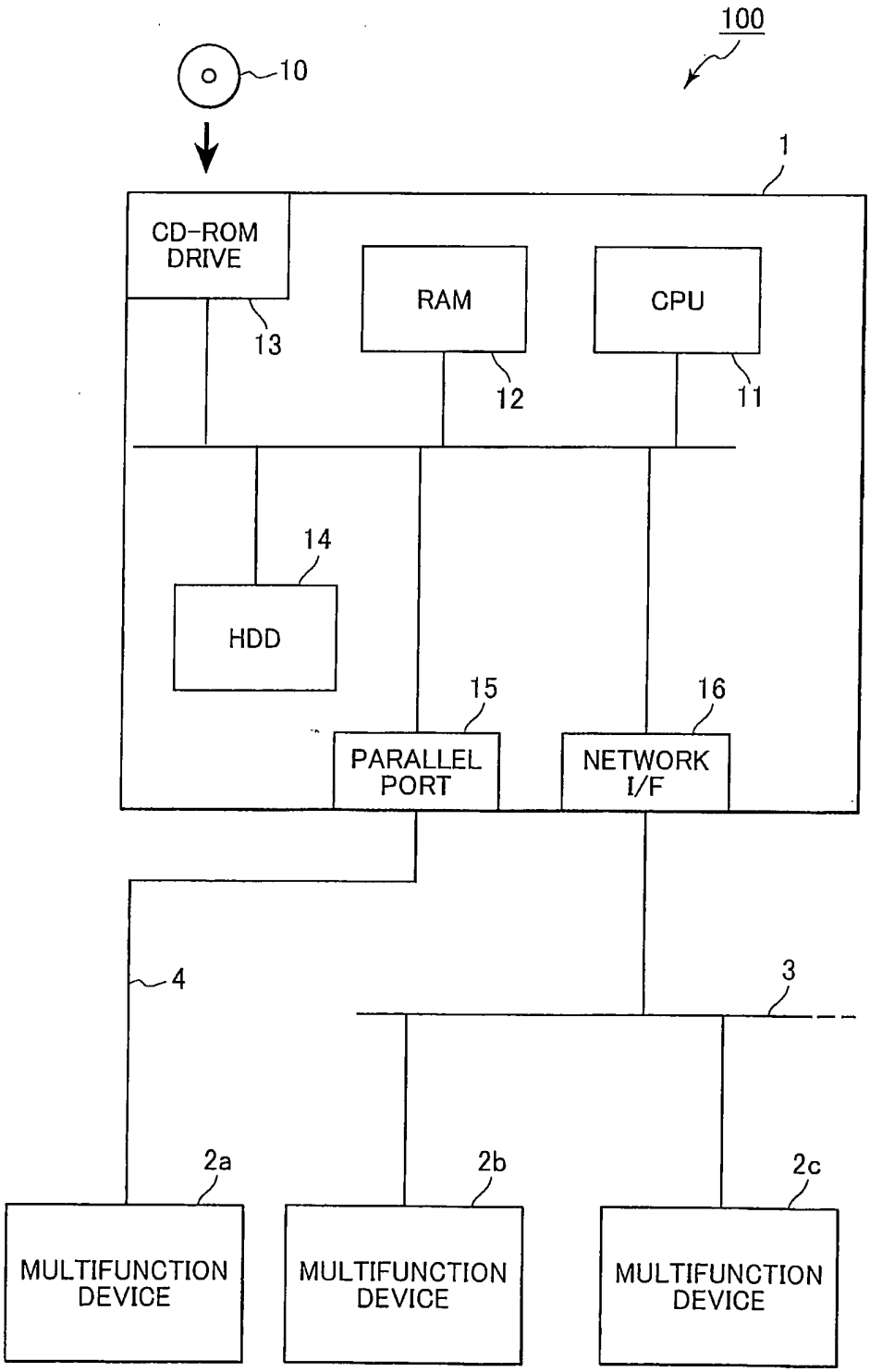


Fig. 2

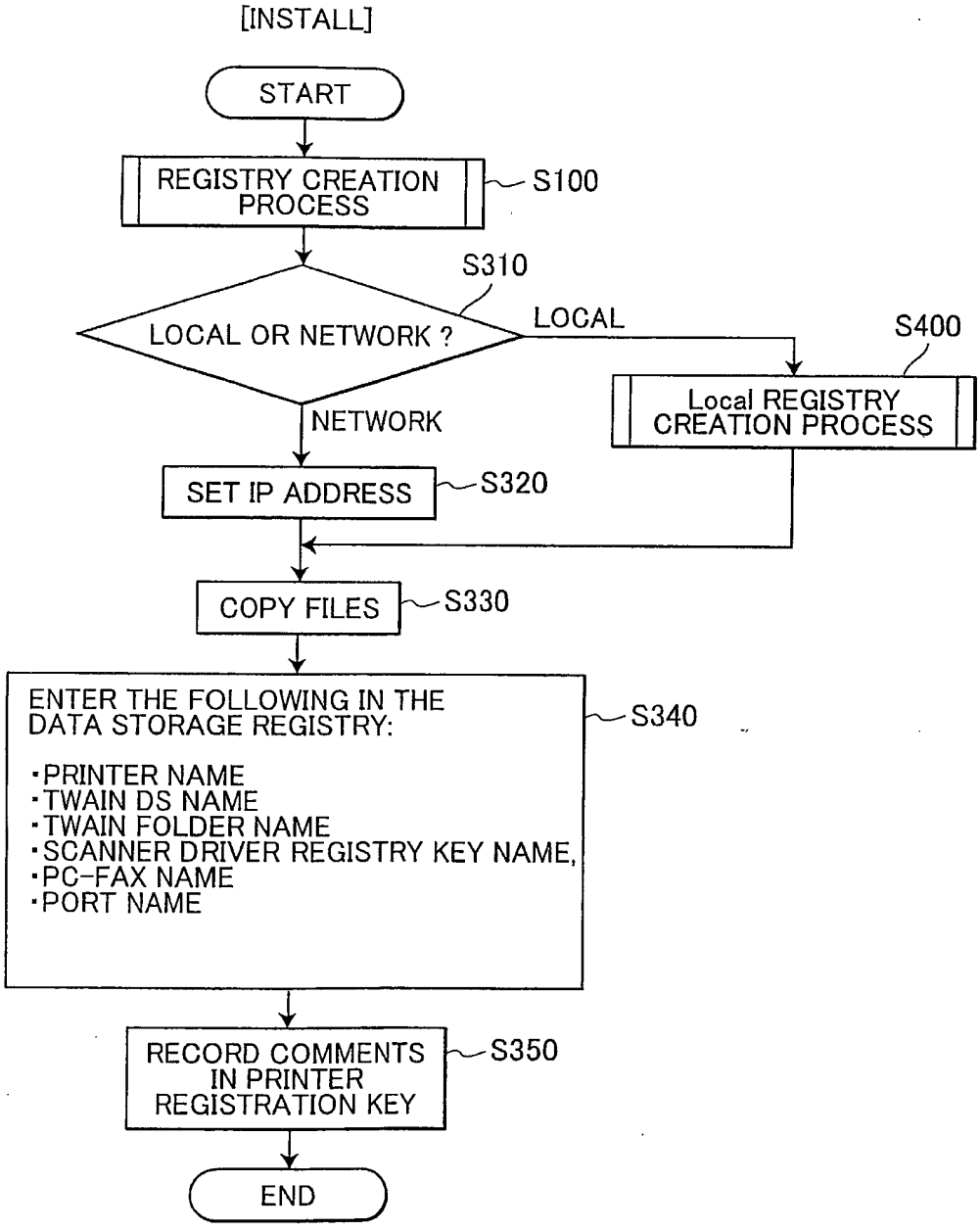

[INSTALL]
START
REGISTRY CREATION PROCESS
S100
S310
LOCAL OR NETWORK ?
LOCAL
S400
Local REGISTRY CREATION PROCESS
NETWORK
SET IP ADDRESS
S320
COPY FILES
S330
ENTER THE FOLLOWING IN THE DATA STORAGE REGISTRY:
・PRINTER NAME
・TWAIN DS NAME
・TWAIN FOLDER NAME
・SCANNER DRIVER REGISTRY KEY NAME,
・PC-FAX NAME
・PORT NAME
S340
RECORD COMMENTS IN PRINTER REGISTRATION KEY
S350
END

Fig. 3

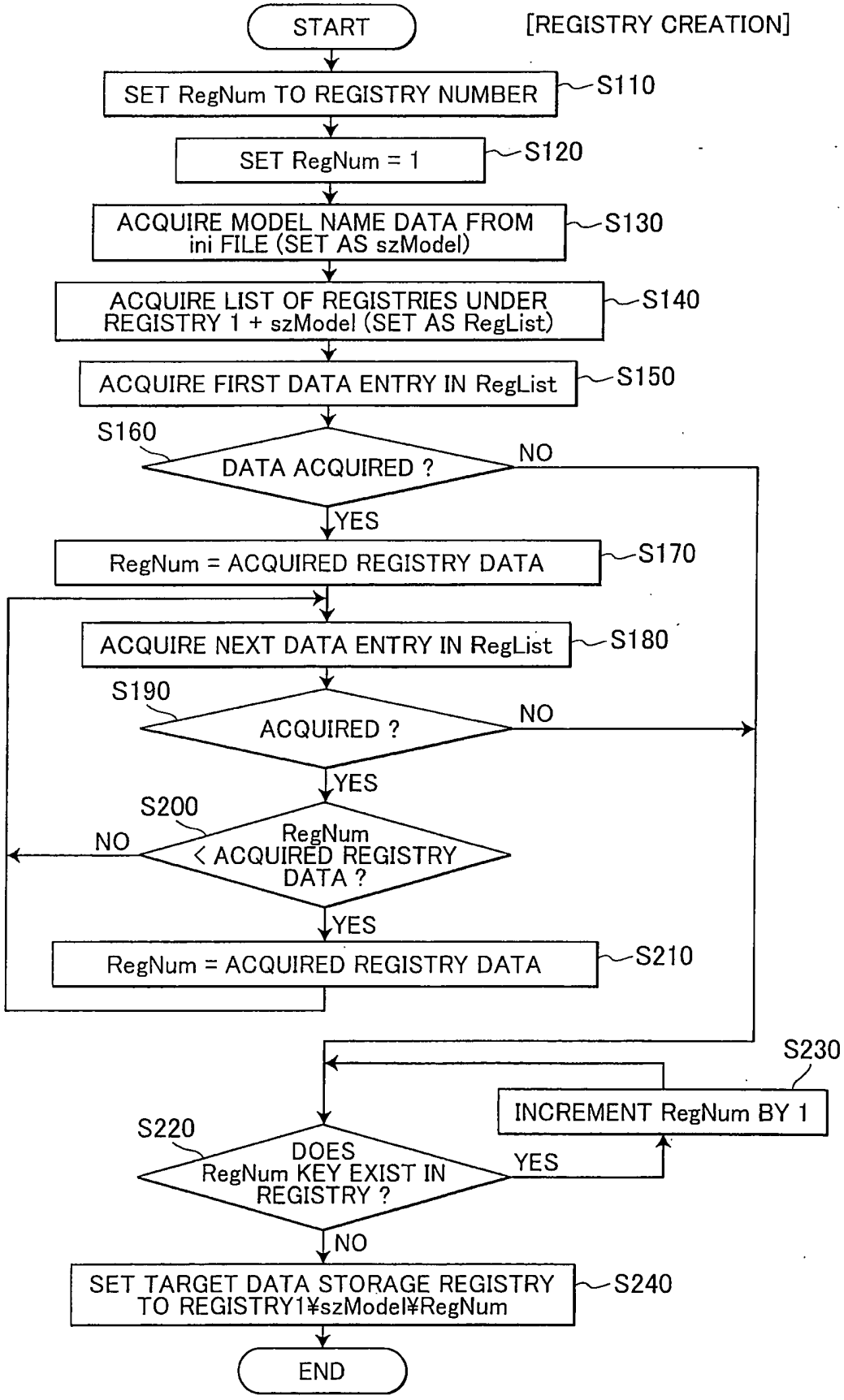
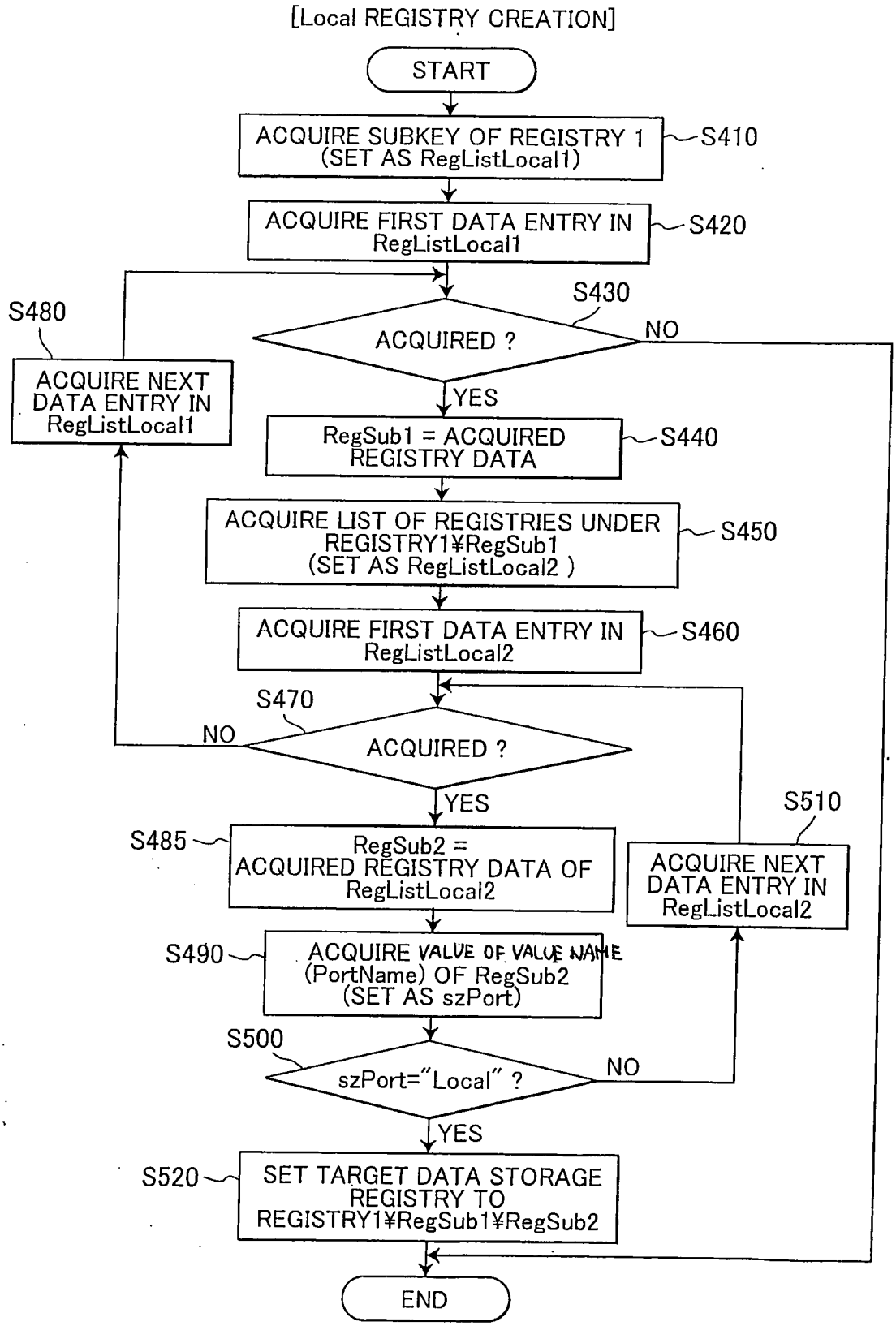

START
[REGISTRY CREATION]
SET RegNum TO REGISTRY NUMBER
S110
SET RegNum = 1
S120
ACQUIRE MODEL NAME DATA FROM ini FILE (SET AS szModel)
S130
ACQUIRE LIST OF REGISTRIES UNDER REGISTRY 1 + szModel (SET AS RegList)
S140
ACQUIRE FIRST DATA ENTRY IN RegList
S150
S160
DATA ACQUIRED ?
NO
YES
RegNum = ACQUIRED REGISTRY DATA
S170
ACQUIRE NEXT DATA ENTRY IN RegList
S180
S190
ACQUIRED ?
NO
YES
S200
RegNum < ACQUIRED REGISTRY DATA ?
NO
YES
RegNum = ACQUIRED REGISTRY DATA
S210
S230
INCREMENT RegNum BY 1
S220
DOES RegNum KEY EXIST IN REGISTRY ?
YES
NO
SET TARGET DATA STORAGE REGISTRY TO REGISTRY1¥szModel¥RegNum
S240
END

Fig. 4


[Local REGISTRY CREATION]
START
ACQUIRE SUBKEY OF REGISTRY 1 (SET AS RegListLocal1)
S410
ACQUIRE FIRST DATA ENTRY IN RegListLocal1
S420
S430
ACQUIRED ?
NO
YES
S480
ACQUIRE NEXT DATA ENTRY IN RegListLocal1
RegSub1 = ACQUIRED REGISTRY DATA
S440
ACQUIRE LIST OF REGISTRIES UNDER REGISTRY1¥RegSub1 (SET AS RegListLocal2)
S450
ACQUIRE FIRST DATA ENTRY IN RegListLocal2
S460
S470
NO
ACQUIRED ?
YES
S510
S485
RegSub2 = ACQUIRED REGISTRY DATA OF RegListLocal2
ACQUIRE NEXT DATA ENTRY IN RegListLocal2
S490
ACQUIRE VALUE OF VALUE NAME (PortName) OF RegSub2 (SET AS szPort)
S500
szPort="Local" ?
NO
YES
S520
SET TARGET DATA STORAGE REGISTRY TO REGISTRY1¥RegSub1¥RegSub2
END

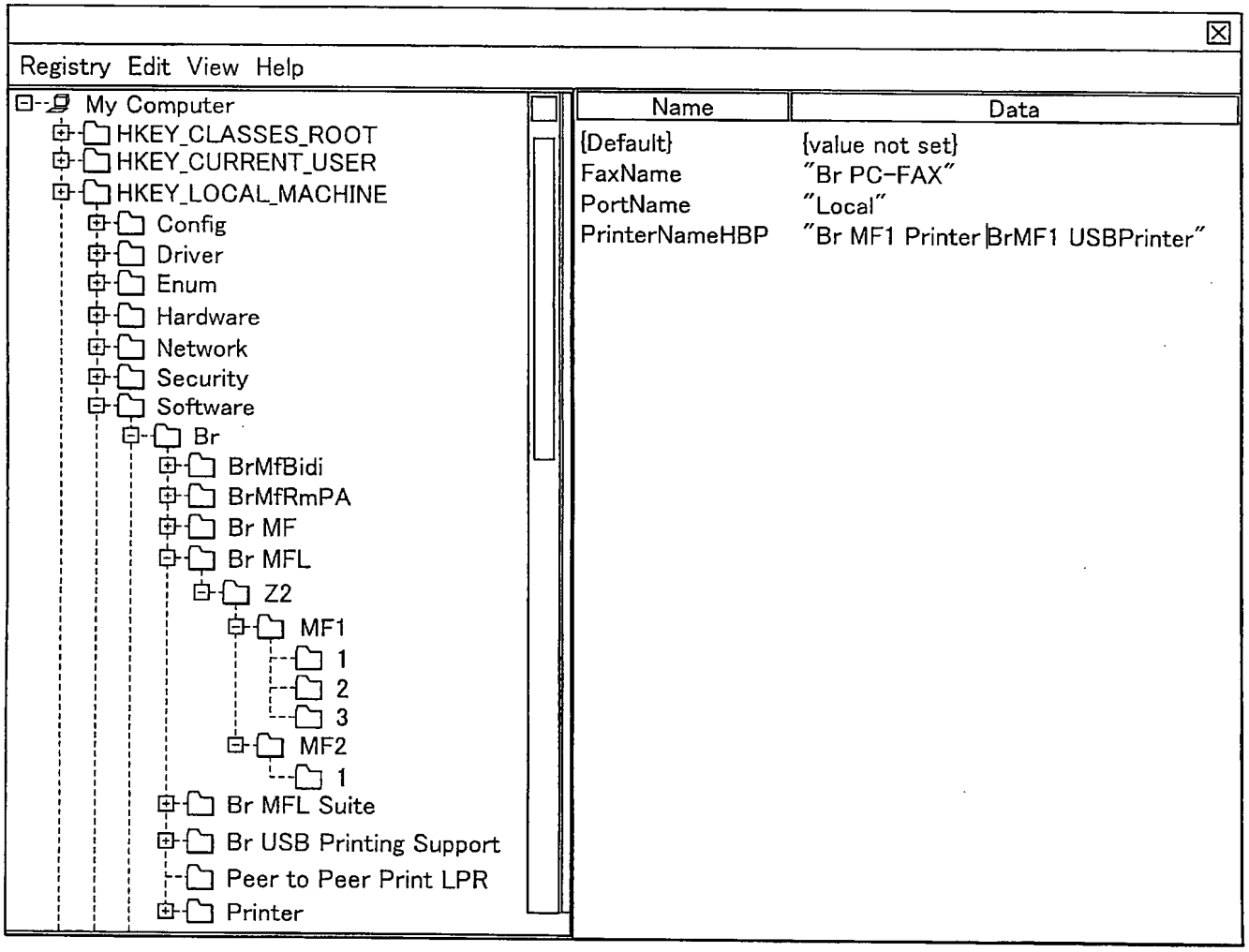

Registry Edit View Help
My Computer
HKEY_CLASSES_ROOT
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
Config
Driver
Enum
Hardware
Network
Security
Software
Br
BrMfBidi
BrMfRmPA
Br MF
Br MFL
Z2
MF1
1
2
3
MF2
1
Br MFL Suite
Br USB Printing Support
Peer to Peer Print LPR
Printer
Name
Data
{Default}
{value not set}
FaxName
"Br PC-FAX"
PortName
"Local"
PrinterNameHBP
"Br MF1 Printer BrMF1 USBPrinter"

Fig. 5

Fig. 6

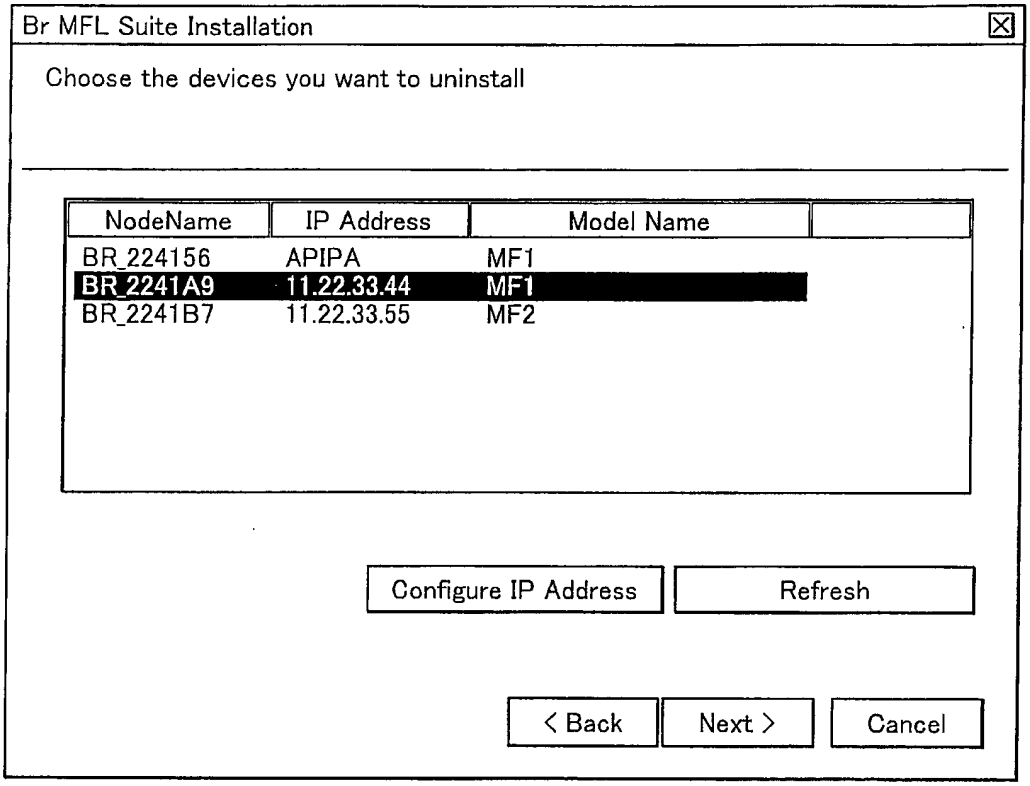

Br MFL Suite Installation
Choose the devices you want to uninstall
NodeName
IP Address
Model Name
BR_224156
APIPA
MF1
BR_2241A9
11.22.33.44
MF1
BR_2241B7
11.22.33.55
MF2
Configure IP Address
Refresh
< Back
Next >
Cancel

Fig. 7

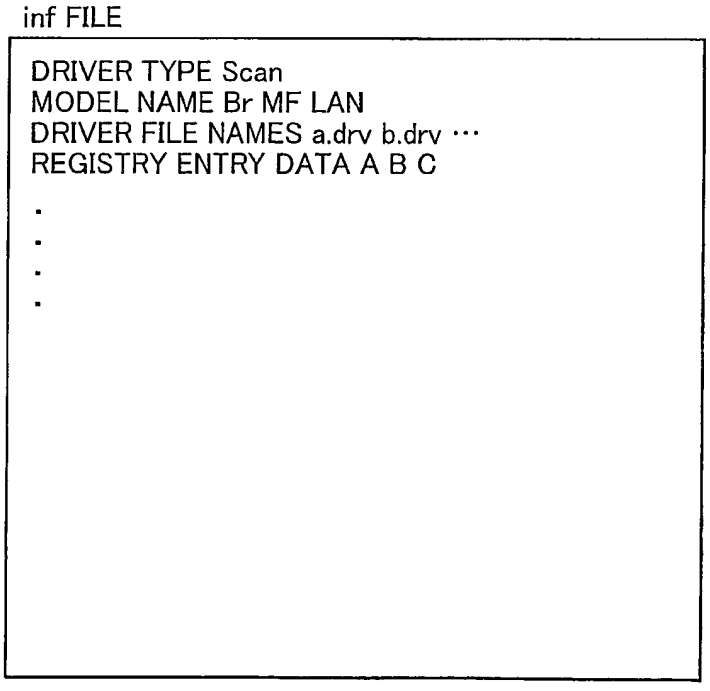

inf FILE
DRIVER TYPE Scan
MODEL NAME Br MF LAN
DRIVER FILE NAMES a.drv b.drv ···
REGISTRY ENTRY DATA A B C

Fig. 8

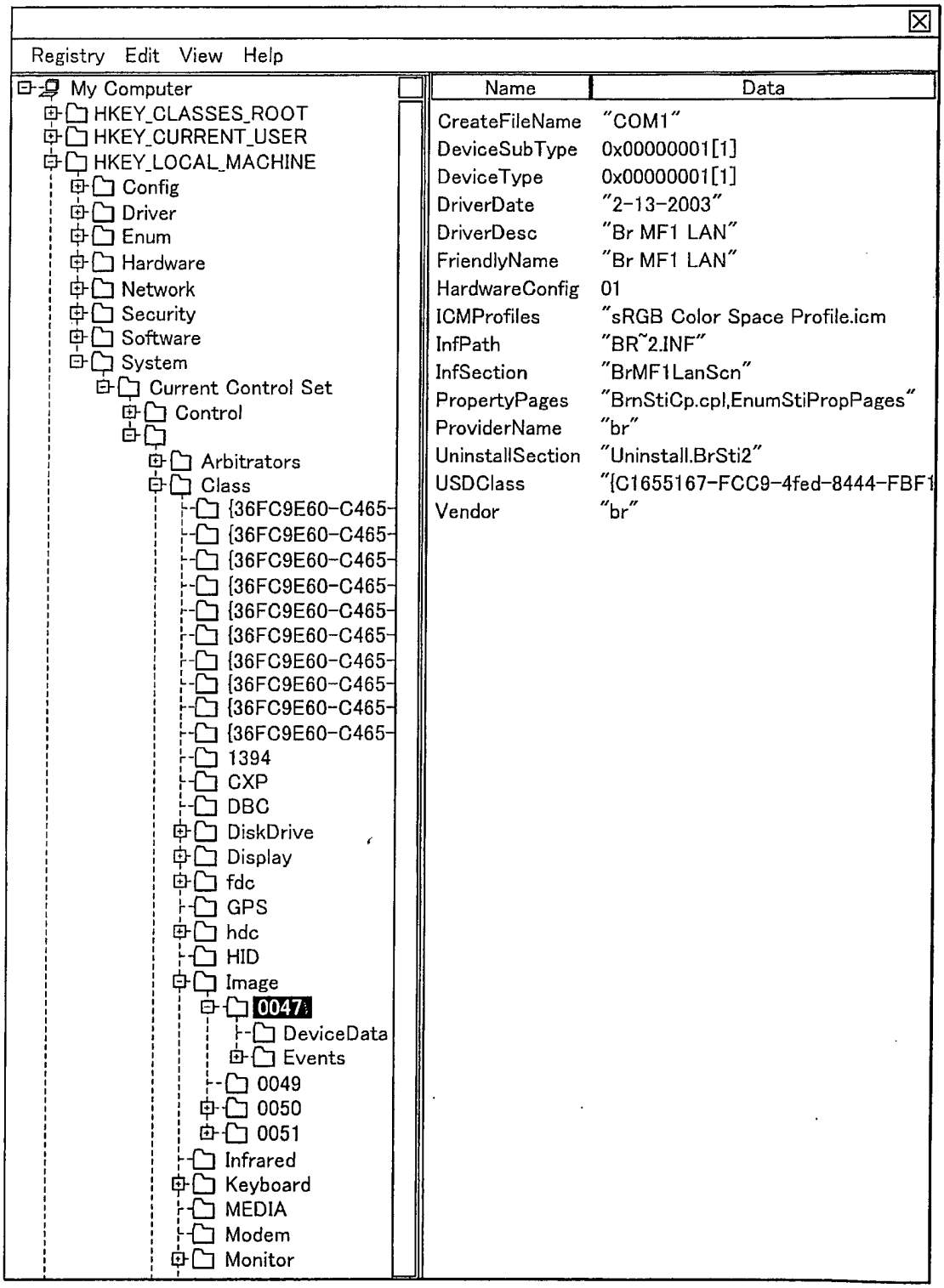

Registry Edit View Help
My Computer
HKEY_CLASSES_ROOT
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
Config
Driver
Enum
Hardware
Network
Security
Software
System
Current Control Set
Control
Arbitrators
Class
{36FC9E60-C465-
{36FC9E60-C465-
{36FC9E60-C465-
{36FC9E60-C465-
{36FC9E60-C465-
{36FC9E60-C465-
{36FC9E60-C465-
{36FC9E60-C465-
{36FC9E60-C465-
{36FC9E60-C465-
1394
CXP
DBC
DiskDrive
Display
fdc
GPS
hdc
HID
Image
0047
DeviceData
Events
0049
0050
0051
Infrared
Keyboard
MEDIA
Modem
Monitor
Name
Data
CreateFileName "COM1"
DeviceSubType 0x00000001[1]
DeviceType 0x00000001[1]
DriverDate "2-13-2003"
DriverDesc "Br MF1 LAN"
FriendlyName "Br MF1 LAN"
HardwareConfig 01
ICMProfiles "sRGB Color Space Profile.icm
InfPath "BR~2.INF"
InfSection "BrMF1LanScn"
PropertyPages "BrnStiCp.cpl,EnumStiPropPages"
ProviderName "br"
UninstallSection "Uninstall.BrSti2"
USDClass "{C1655167-FCC9-4fed-8444-FBF1
Vendor "br"

Fig. 9

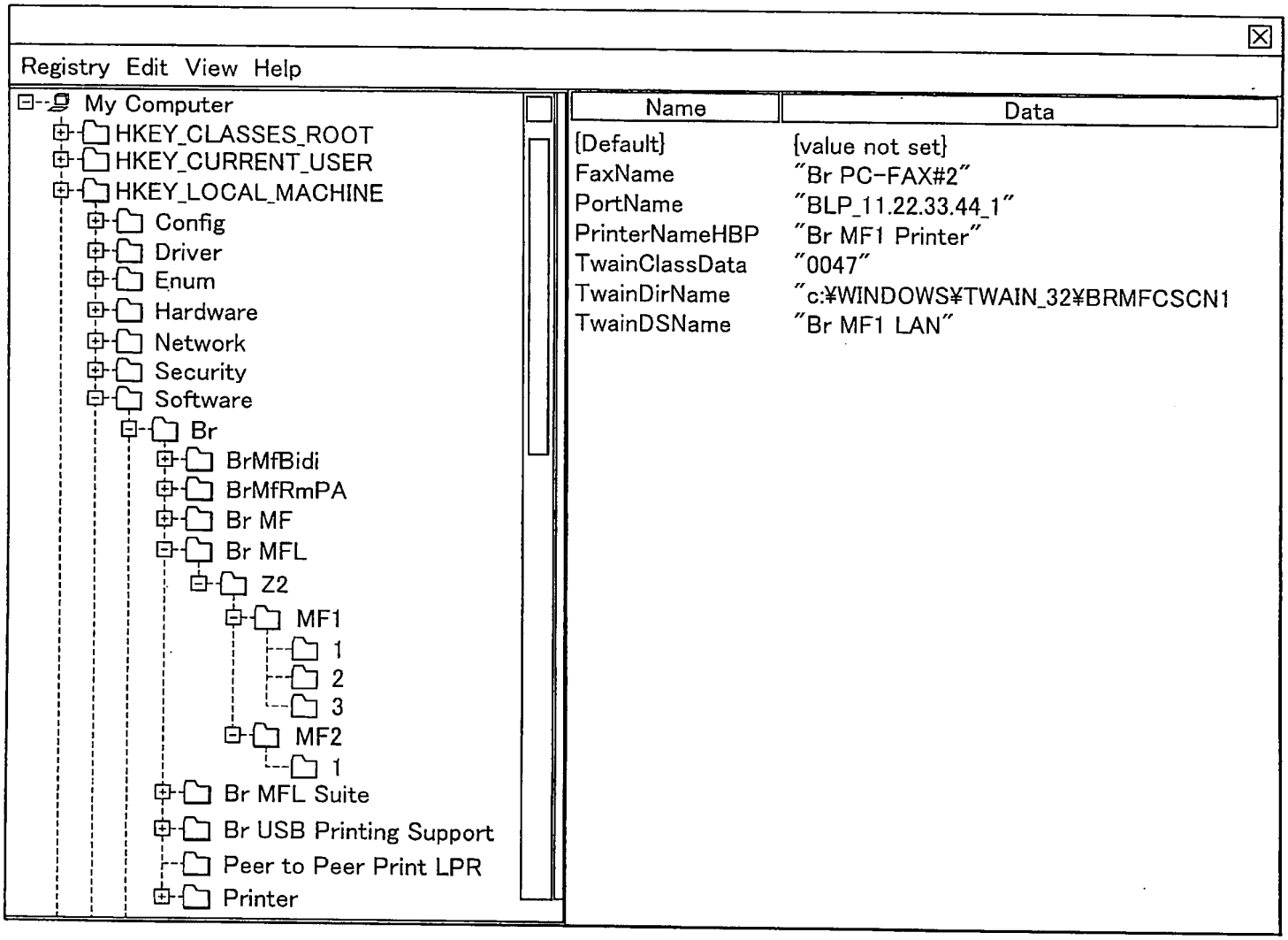

Registry Edit View Help
My Computer
HKEY_CLASSES_ROOT
HKEY_CURRENT_USER
HKEY_LOCAL_MACHINE
Config
Driver
Enum
Hardware
Network
Security
Software
Br
BrMfBidi
BrMfRmPA
Br MF
Br MFL
Z2
MF1
1
2
3
MF2
1
Br MFL Suite
Br USB Printing Support
Peer to Peer Print LPR
Printer
Name
Data
{Default}
{value not set}
FaxName
"Br PC-FAX#2"
PortName
"BLP_11.22.33.44_1"
PrinterNameHBP
"Br MF1 Printer"
TwainClassData
"0047"
TwainDirName
"c:¥WINDOWS¥TWAIN_32¥BRMFCSCN1
TwainDSName
"Br MF1 LAN"

Fig. 10

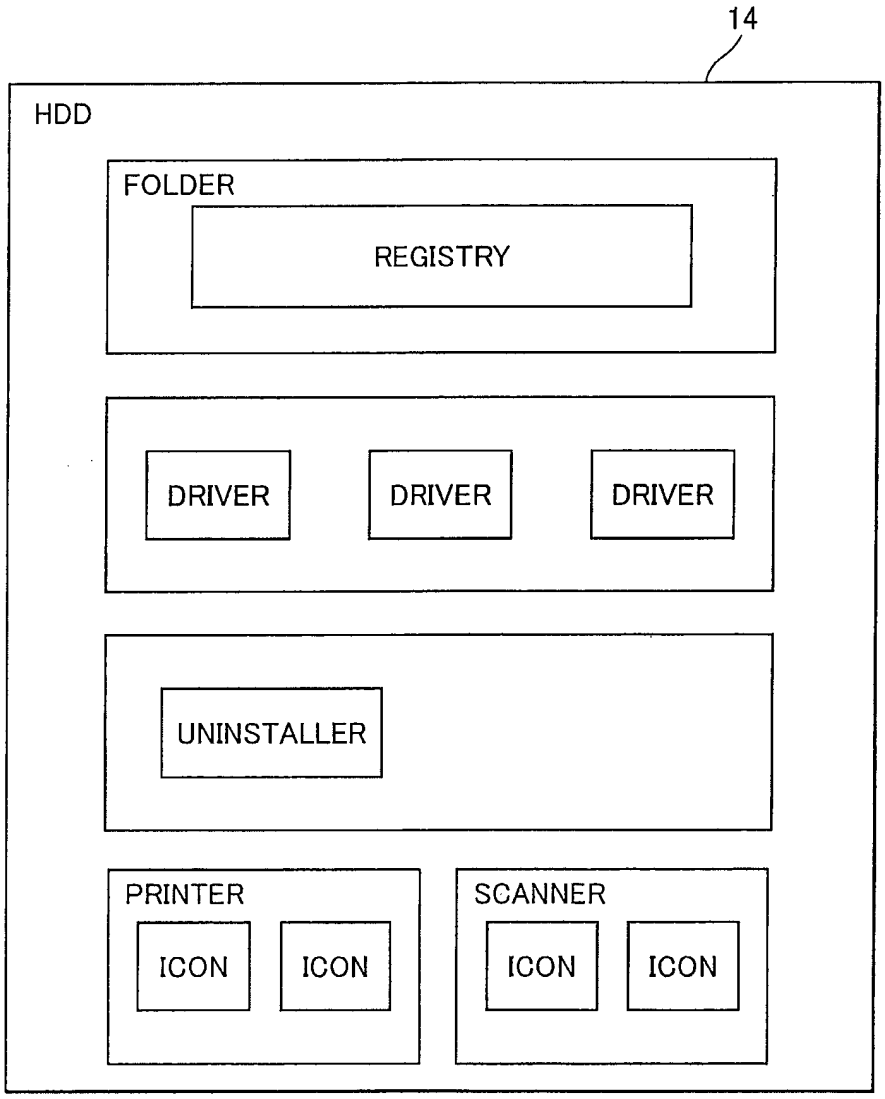

14
HDD
FOLDER
REGISTRY
DRIVER
DRIVER
DRIVER
UNINSTALLER
PRINTER
ICON
ICON
SCANNER
ICON
ICON

Fig.11

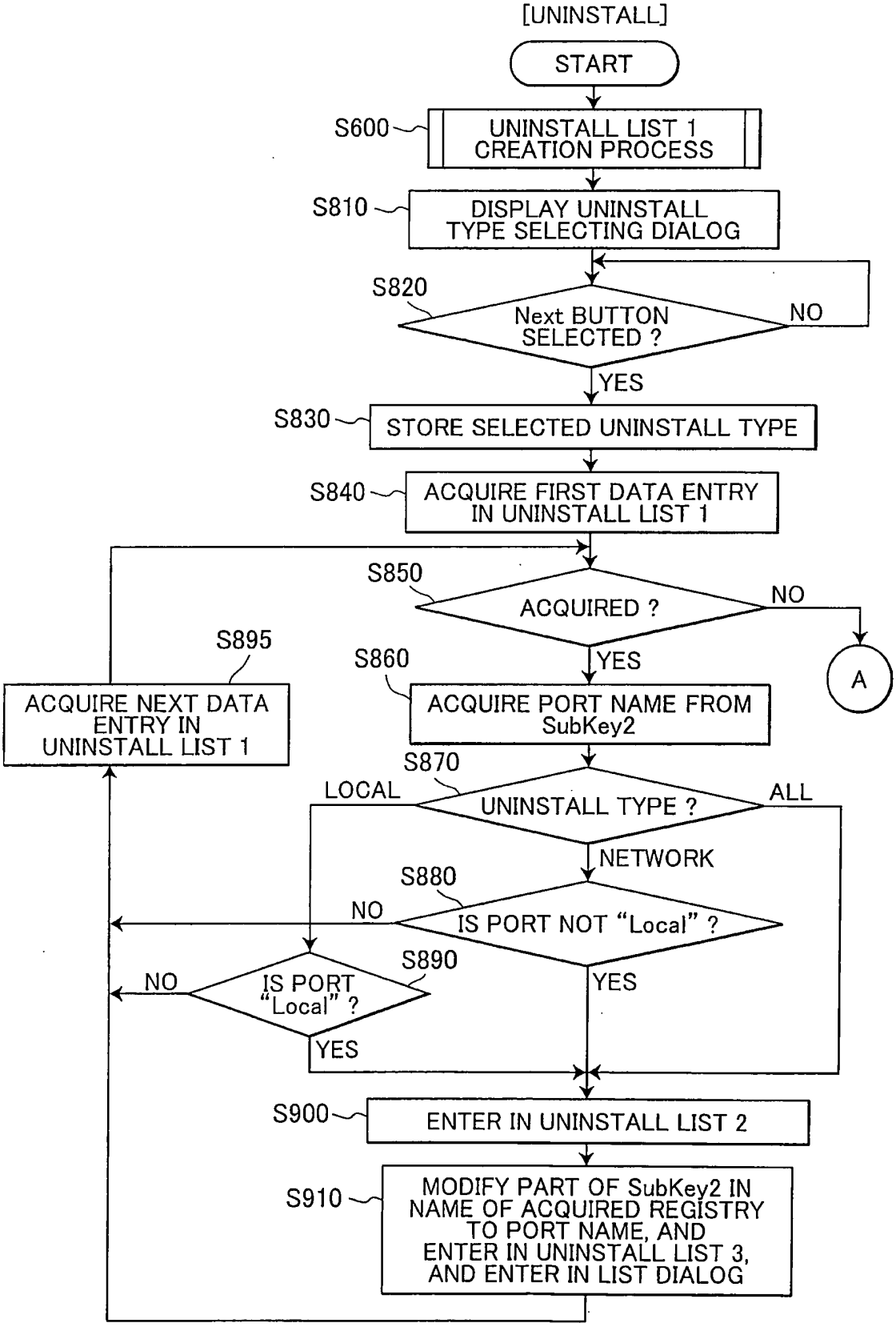

[UNINSTALL]
START
S600
UNINSTALL LIST 1 CREATION PROCESS
S810
DISPLAY UNINSTALL TYPE SELECTING DIALOG
S820
Next BUTTON SELECTED ?
NO
YES
S830
STORE SELECTED UNINSTALL TYPE
S840
ACQUIRE FIRST DATA ENTRY IN UNINSTALL LIST 1
S850
ACQUIRED ?
NO
YES
A
S860
ACQUIRE PORT NAME FROM SubKey2
S870
LOCAL
UNINSTALL TYPE ?
ALL
NETWORK
S880
NO
IS PORT NOT "Local" ?
YES
S890
NO
IS PORT "Local" ?
YES
S895
ACQUIRE NEXT DATA ENTRY IN UNINSTALL LIST 1
S900
ENTER IN UNINSTALL LIST 2
S910
MODIFY PART OF SubKey2 IN NAME OF ACQUIRED REGISTRY TO PORT NAME, AND ENTER IN UNINSTALL LIST 3, AND ENTER IN LIST DIALOG

Fig. 12

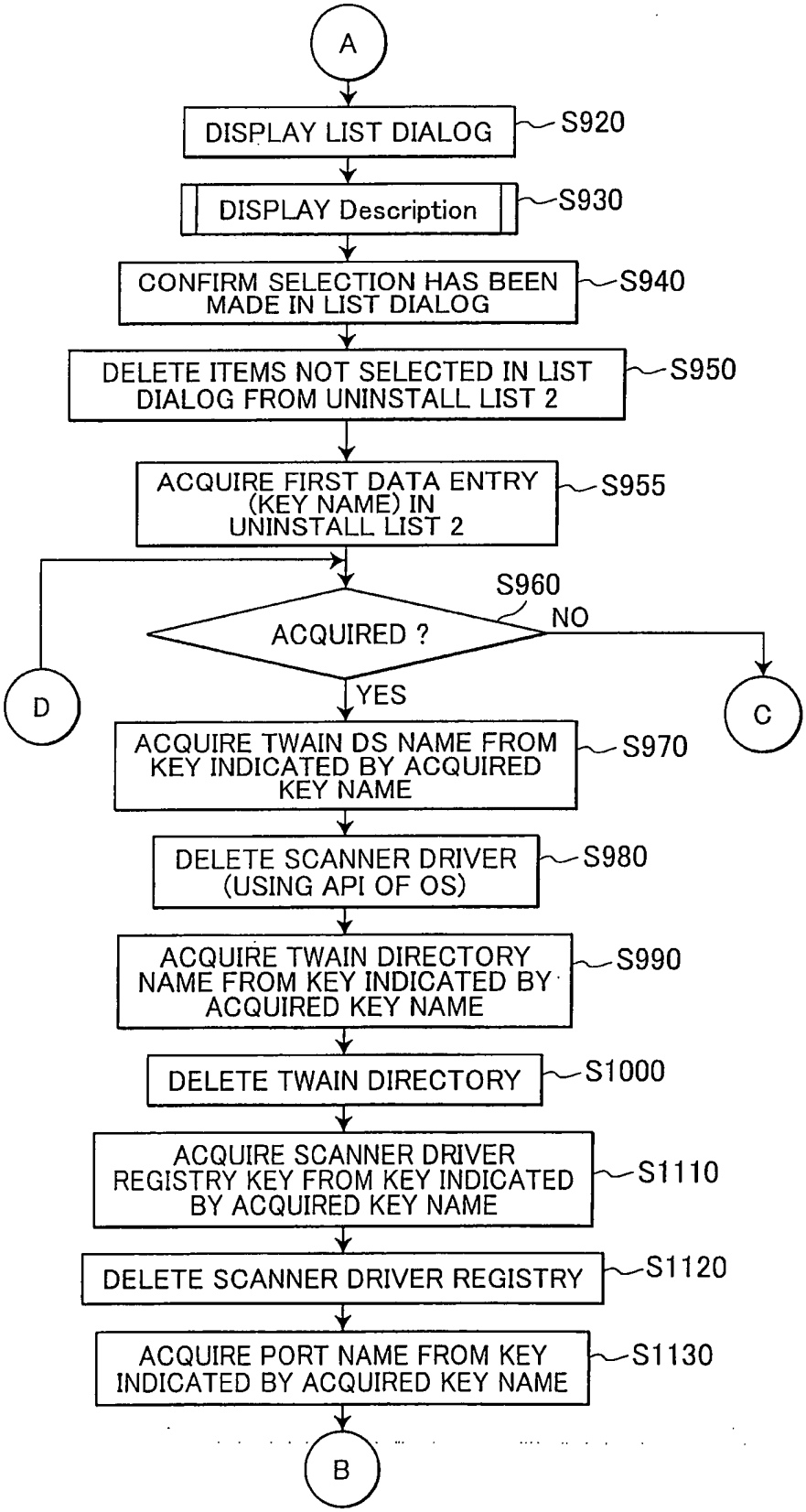

A
DISPLAY LIST DIALOG
S920
DISPLAY Description
S930
CONFIRM SELECTION HAS BEEN MADE IN LIST DIALOG
S940
DELETE ITEMS NOT SELECTED IN LIST DIALOG FROM UNINSTALL LIST 2
S950
ACQUIRE FIRST DATA ENTRY (KEY NAME) IN UNINSTALL LIST 2
S955
S960
ACQUIRED ?
NO
YES
D
C
ACQUIRE TWAIN DS NAME FROM KEY INDICATED BY ACQUIRED KEY NAME
S970
DELETE SCANNER DRIVER (USING API OF OS)
S980
ACQUIRE TWAIN DIRECTORY NAME FROM KEY INDICATED BY ACQUIRED KEY NAME
S990
DELETE TWAIN DIRECTORY
S1000
ACQUIRE SCANNER DRIVER REGISTRY KEY FROM KEY INDICATED BY ACQUIRED KEY NAME
S1110
DELETE SCANNER DRIVER REGISTRY
S1120
ACQUIRE PORT NAME FROM KEY INDICATED BY ACQUIRED KEY NAME
S1130
B

Fig. 13

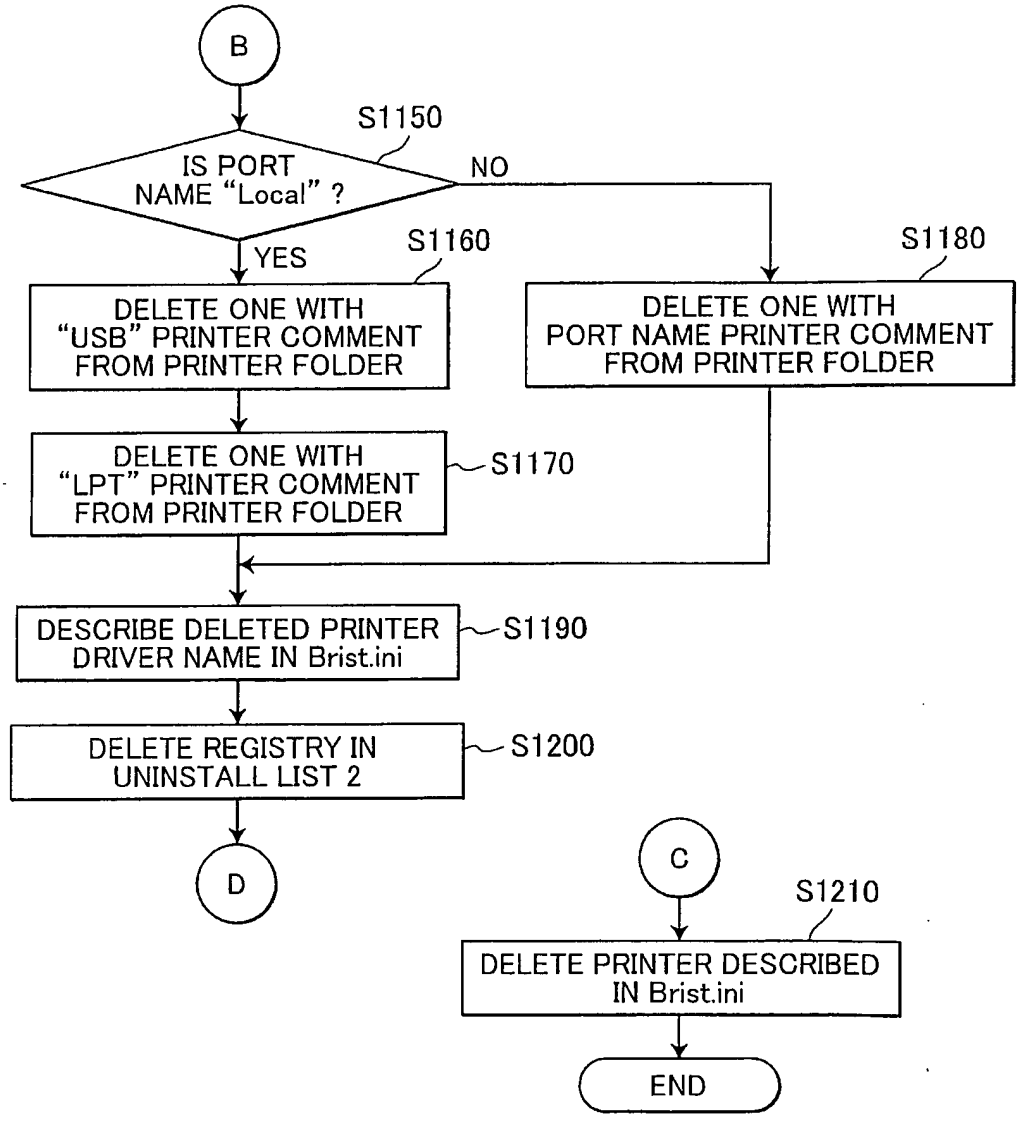
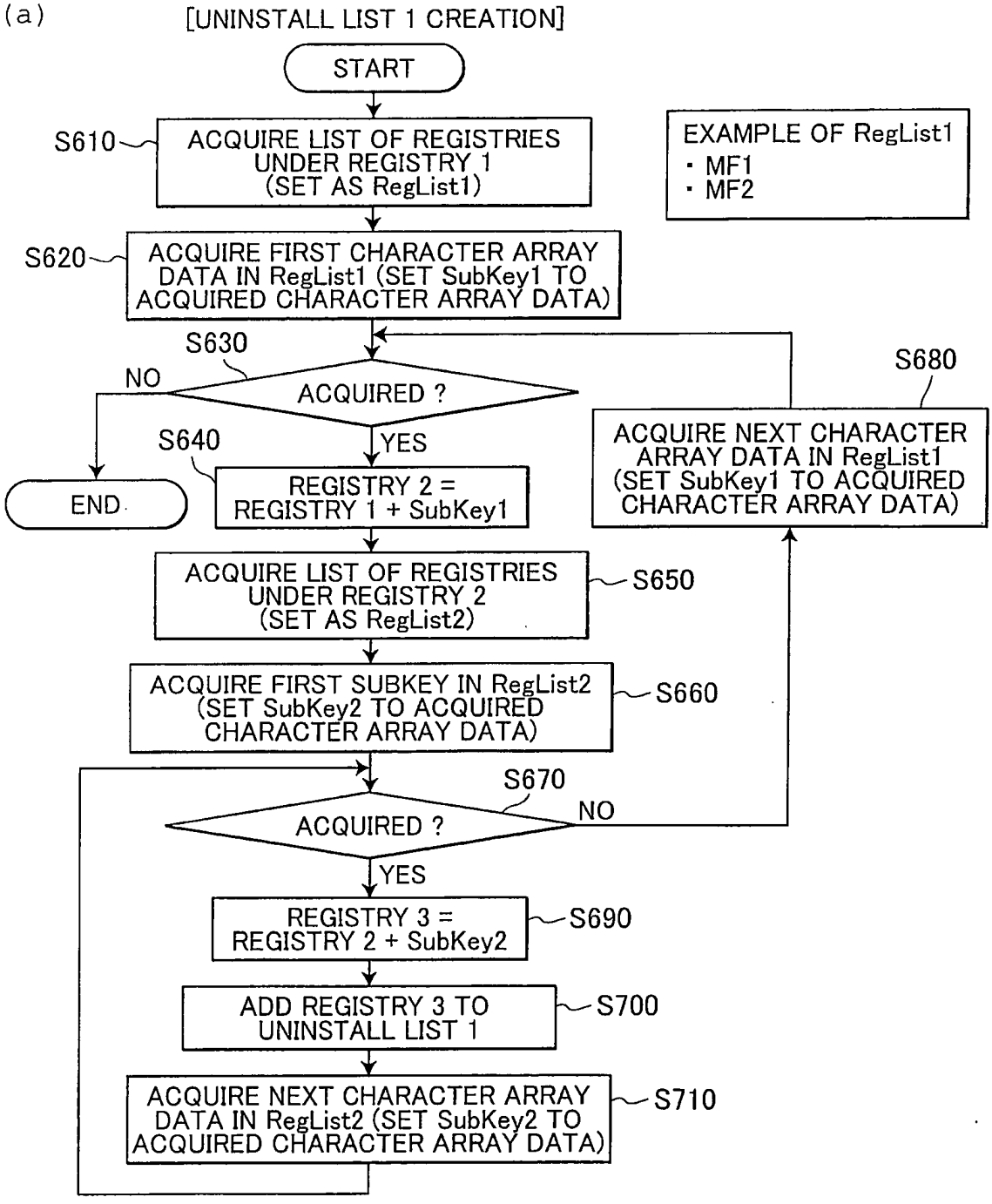

B
S1150
IS PORT NAME "Local" ?
NO
YES
S1160
DELETE ONE WITH "USB" PRINTER COMMENT FROM PRINTER FOLDER
S1180
DELETE ONE WITH PORT NAME PRINTER COMMENT FROM PRINTER FOLDER
DELETE ONE WITH "LPT" PRINTER COMMENT FROM PRINTER FOLDER
S1170
DESCRIBE DELETED PRINTER DRIVER NAME IN Brist.ini
S1190
DELETE REGISTRY IN UNINSTALL LIST 2
S1200
D
C
S1210
DELETE PRINTER DESCRIBED IN Brist.ini
END

Fig. 14

(a)



(b)

REGISTRY 1
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2

REGISTRY 2
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1

REGISTRY 3
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥1

Fig. 15

(a) UNINSTALL LIST 1
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥1
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥2
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥3
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF2¥1

(b)

EXAMPLE IF USER SELECTS "Local" AS UNINSTALL TYPE AND "Local" IN UNINSTALL LIST IS "2"

UNINSTALL LIST 2
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥2

UNINSTALL LIST 3
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥Local

(c) EXAMPLE IF "ALL" IS SELECTED AS UNINSTALL TYPE

UNINSTALL LIST 2
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥1
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥2
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥3
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF2¥1

UNINSTALL LIST 3
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥Local
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥BLP_11.22.33.44_1
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF1¥BLP_11.22.33.45_1
HKEY_LOCAL_MACHINE¥Software¥Br¥Br MFL¥Z2¥MF2¥BLP_11.22.33.55_1

Fig. 16

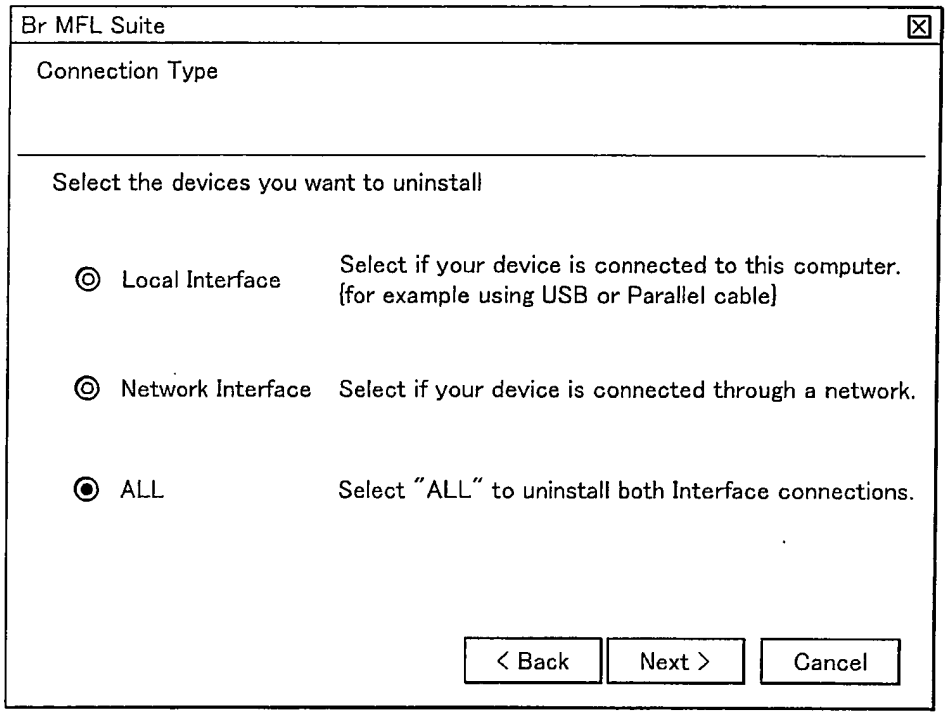

Br MFL Suite
Connection Type
Select the devices you want to uninstall
Local Interface
Select if your device is connected to this computer. (for example using USB or Parallel cable)
Network Interface
Select if your device is connected through a network.
ALL
Select "ALL" to uninstall both Interface connections.
< Back
Next >
Cancel

Fig. 17

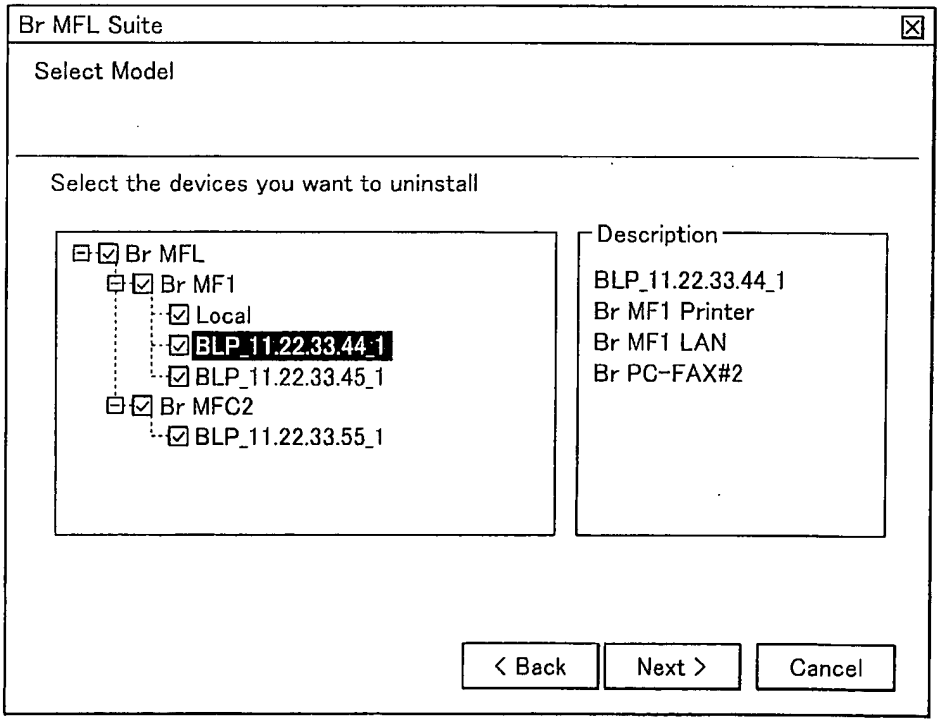

Br MFL Suite
Select Model
Select the devices you want to uninstall
Br MFL
Br MF1
Local
BLP_11.22.33.44_1
BLP_11.22.33.45_1
Br MFC2
BLP_11.22.33.55_1
Description
BLP_11.22.33.44_1
Br MF1 Printer
Br MF1 LAN
Br PC-FAX#2
< Back
Next >
Cancel

Fig. 18

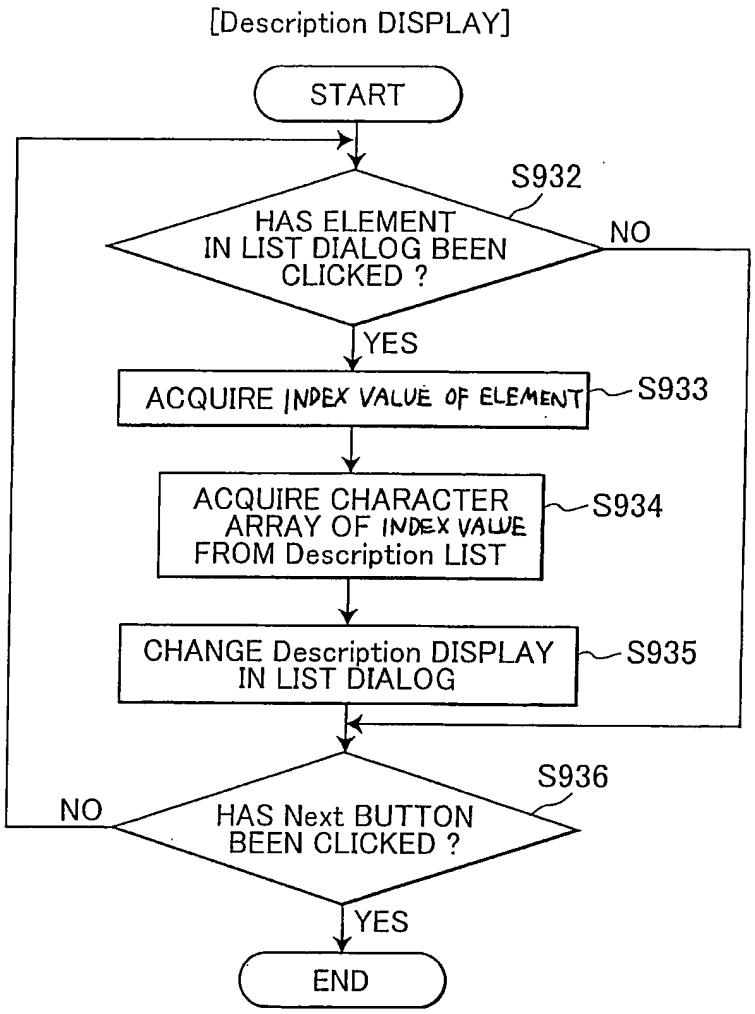

[Description DISPLAY]
START
S932
HAS ELEMENT IN LIST DIALOG BEEN CLICKED ?
NO
YES
ACQUIRE INDEX VALUE OF ELEMENT
S933
ACQUIRE CHARACTER ARRAY OF INDEX VALUE FROM Description LIST
S934
CHANGE Description DISPLAY IN LIST DIALOG
S935
S936
NO
HAS Next BUTTON BEEN CLICKED ?
YES
END

Fig. 19

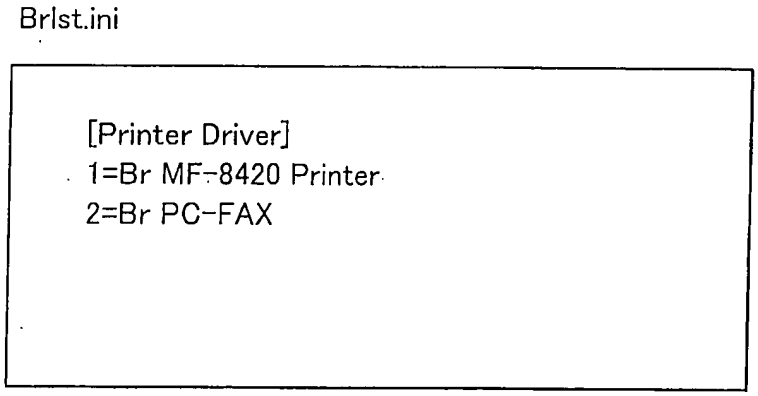
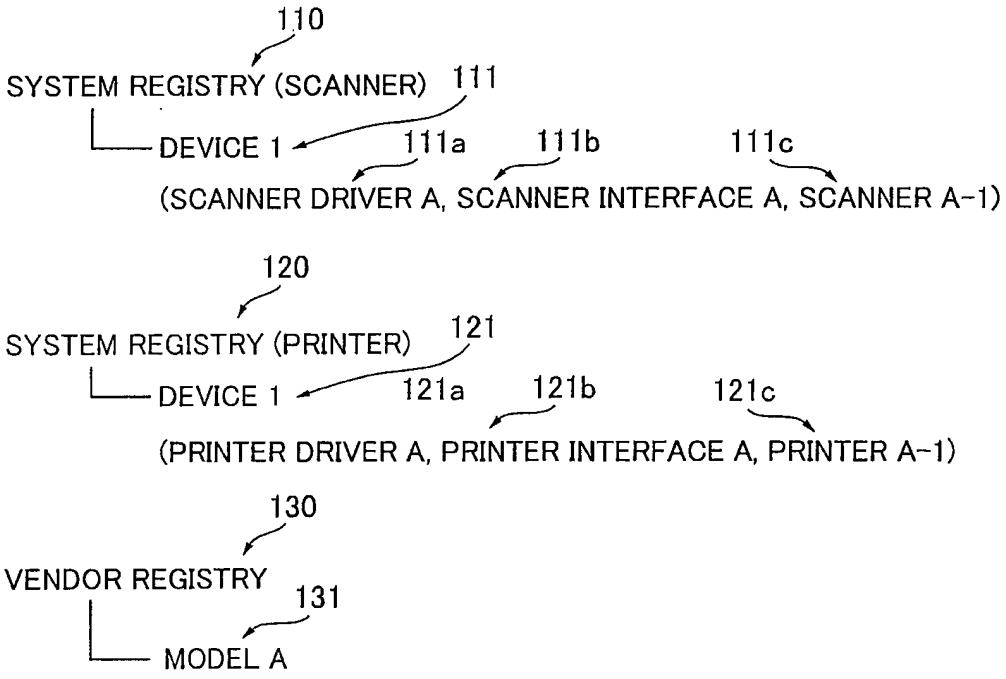

BrIst.ini
[Printer Driver]
1=Br MF-8420 Printer
2=Br PC-FAX

Fig. 20

(a)


110
SYSTEM REGISTRY (SCANNER)
111
DEVICE 1
111a
111b
111c
(SCANNER DRIVER A, SCANNER INTERFACE A, SCANNER A-1)
120
SYSTEM REGISTRY (PRINTER)
121
DEVICE 1
121a
121b
121c
(PRINTER DRIVER A, PRINTER INTERFACE A, PRINTER A-1)
130
VENDOR REGISTRY
131
MODEL A

(b)

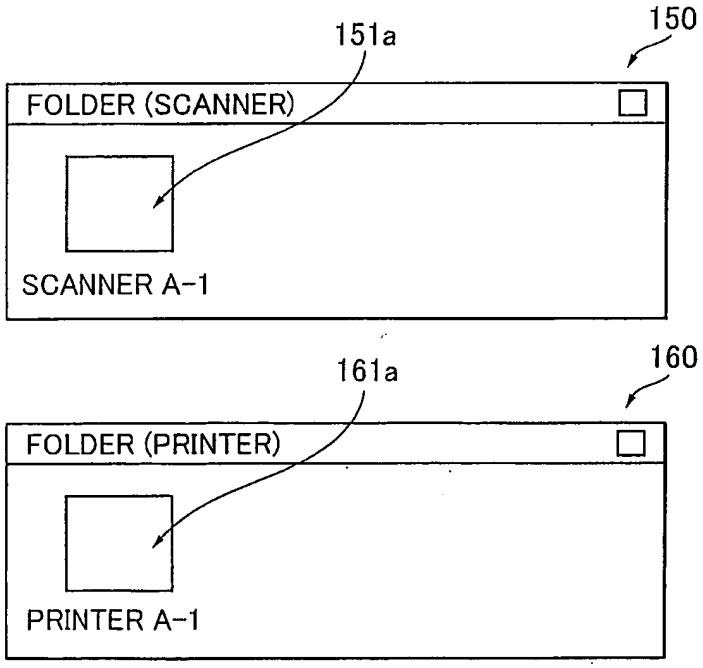
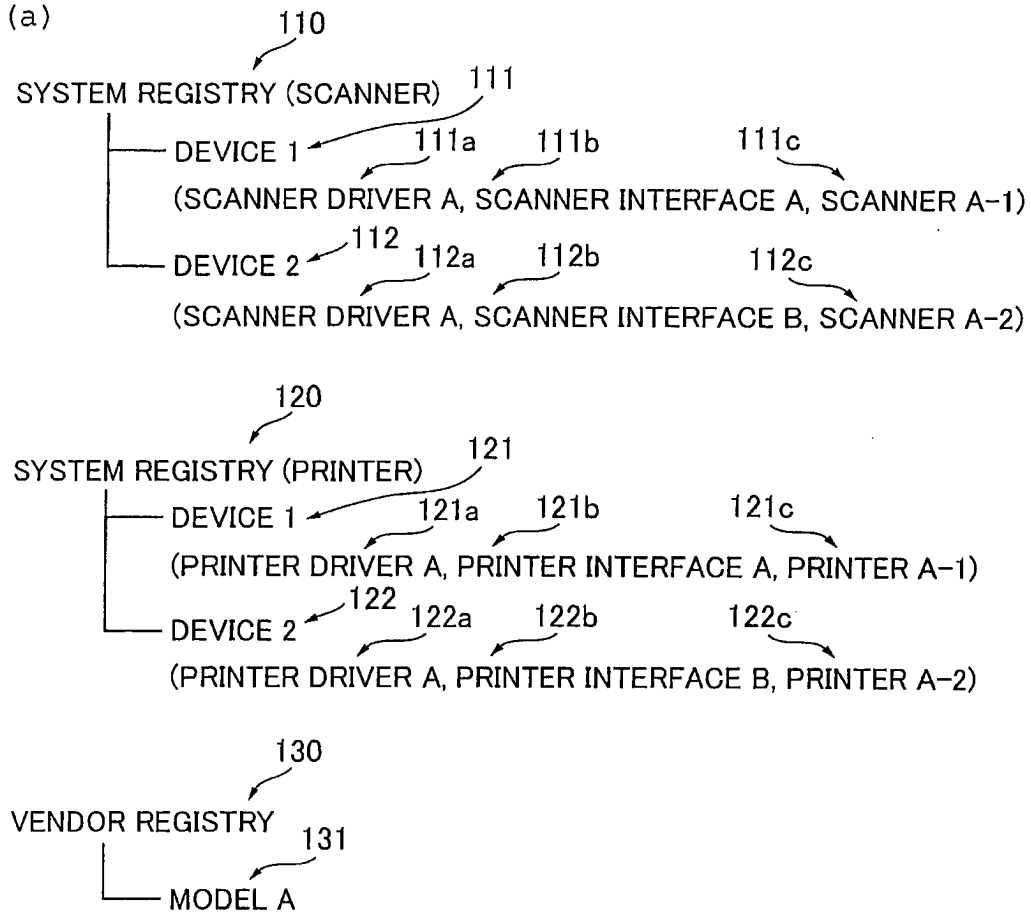
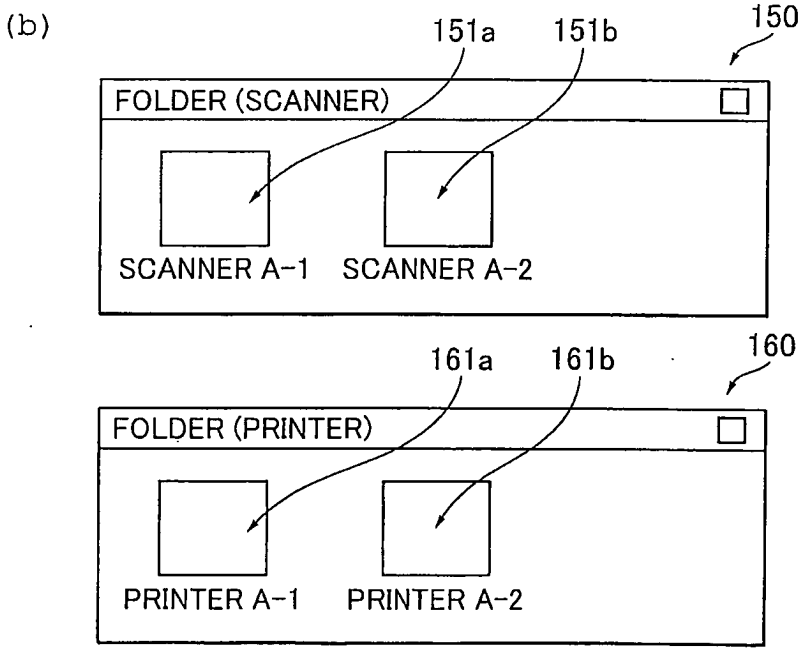

151a
150
FOLDER (SCANNER)
SCANNER A-1
161a
160
FOLDER (PRINTER)
PRINTER A-1

Fig. 21


(a)
110
SYSTEM REGISTRY (SCANNER)
111
DEVICE 1
111a
111b
111c
(SCANNER DRIVER A, SCANNER INTERFACE A, SCANNER A-1)
112
DEVICE 2
112a
112b
112c
(SCANNER DRIVER A, SCANNER INTERFACE B, SCANNER A-2)
120
SYSTEM REGISTRY (PRINTER)
121
DEVICE 1
121a
121b
121c
(PRINTER DRIVER A, PRINTER INTERFACE A, PRINTER A-1)
122
DEVICE 2
122a
122b
122c
(PRINTER DRIVER A, PRINTER INTERFACE B, PRINTER A-2)
130
VENDOR REGISTRY
131
MODEL A


(b)
151a
151b
150
FOLDER (SCANNER)
SCANNER A-1
SCANNER A-2
161a
161b
160
FOLDER (PRINTER)
PRINTER A-1
PRINTER A-2